Multi-Level Modeling for Distribution Centers

Gerrit Muller (Buskerud University College, Kongsberg, Norway),

Roelof Hamberg (Embedded Systems Institute, Eindhoven, Nederland)[1]

Case Study Elements

- **Fundamental essence**: the application of multi-level models to guide the development of enterprise-level systems of systems (SoSs).
- **Topical relevance**: the approach is critically relevant in the sense that emergent behavior appearing at higher levels in systems is often unexpected and not as intended.
- **Domain**: Industry, logistics domain
- **Country**: The Netherlands
- **Stakeholders**: enterprise architects for industrial SoSs
- **Primary Insights**: A federation of connected explorative models at different levels of abstraction is necessary to cope with the complexity at the SoS levels of logistical supply chains, in general, and the associated distribution centers, in particular. This is especially relevant with respect to the aspects of dynamics and uncertainties.

[1] This work has been carried out as part of the FALCON project under the responsibility of the Embedded Systems Institute with Vanderlande Industries as the carrying industrial partner. This project is partially supported by the Netherlands Ministry of Economic Affairs under the Embedded Systems Institute (BSIK03021) program.

Keywords

goods flow simulation, model-based engineering, multiple levels of abstraction, system-of-systems

Abstract

Distribution centers comprise many layers of decomposition in systems, subsystems, and components, while they are part of larger enterprises. The relative independence of the development of all these layers results in emergent behavior that confronts systems engineers with the challenge to keep a comprehensive overview. A short introduction of the logistic domain leads to a summary of the essential characteristics of distribution center systems and their relation to the emergence of behavior. A series of modeling examples is offered to illustrate how explorative models can be applied to govern the complexity of developing these systems. In order to bridge the multiple levels of system scope, these models are connected through essential parameters that carry the de facto abstractions of subsystems to their representation at a higher level.

Glossary

ACP = Automated Case Picking

AIP = Automatic Item Picking

B = total buffer space in the system

CAD = computer aided design

DC = distribution center

DOF = degrees of freedom

ED = enterprise dynamics

FIFO = First-In-First-Out

SoS = system of systems

Background

Context

This case study is based on the Falcon research project on warehouses of the future (see [Hamberg 2011] and [Falcon 2011]). Vanderlande Industries, the Embedded Systems Institute and several partners cooperated in this research project with the objective to "find and develop efficient means to be able to analyze, design, and implement layered systems which shall comply with stringent requirements on performance, reliability, cost, development time…". The overall challenge was to combine top-down and bottom-up reasoning of business developments, system-level solutions, subsystem feasibility estimations, and the development of promising new technologies. One of the areas of research was the application of modeling to achieve the goal, dealing with this challenge. As one could have anticipated, the achieved project results not only are applicable for the future, but also for current developments that take place in the domain of warehousing.

The structure of this case study is as follows. First, the domain of distribution centers, the logistics supply chain, and the role of distribution center (DC) suppliers are described in a global way. This provides the context for understanding the systems engineering challenges, which are addressed in a later section. Customer-level scenarios facilitate a first exploration of these challenges by making the problems and possible solutions more tangible. A number of modeling examples are discussed next to amplify the insights at different levels of the SoSs, systems, and

subsystems. After a reflection on the utility of these models, the case study is concluded with some guidelines for systems engineers entering the era of SoSs.

Relevant Definitions

In logistic chains, distribution centers (DCs) serve as decoupling and buffering points when transporting goods from suppliers (production centers, such as factories) to outlets (such as retail shops).

An *item* denotes the smallest defined unit in the logistic chain. We use the term *case* for the immediate packaging around a set of goods; it contains one or multiple items. A case can be a carton box or plastic foil. For example, a carton box containing one or multiple identical items, or a number of bottles shrink-wrapped with plastic foil. Cases are stacked on *pallets*, which can be packed in foil, too. Items can also be kept together more loosely in so-called *totes*. Totes contain either multiple identical items or different items depending on its role in the logistic process, i.e., efficient, temporary storage of individual items, or a way to do collect items that belong to a single customer order. Totes have a uniform shape and size in order to optimize their handling by logistic equipment. They are stacked on pallets or *dollies* for efficiency during transport outside the DC.

Pertaining Theories

A term that is frequently used in the SoS context is *emergence*. The aggregation of systems may show behavior or properties that emerge, i.e. that *cannot be localized to a single independently acting constituent or to a small constant number of constituents. Emergent properties arise from the cumulative effects of the local actions and neighbor interactions of many autonomous entities* (quoted from [Fisher 2006]). Emergent properties can be desired or undesired and expected or

unexpected by creators and integrators [Kopetz 2011].

Models with *higher levels of abstraction* (that better match the higher levels of complexity of SoSs [Ashby 2001]) are a means to cope with increased complexity. A higher level of abstraction means to represent essential properties in a fundamental way without losing their characteristic dependencies, and to leave out non-essential properties altogether. System-level properties that are known to relate to emergent behavior require extra caution in this abstraction process, as the emergent properties may derive from lower level actions and interactions.

Systems engineers must find proper levels and ways of abstraction. They need to relate models at various abstraction levels to reason about specification, design, and engineering challenges. We hope that discussion of several examples will illustrate the potential utility of modeling for SoSs.

We advocate that reasoning about SoSs needs to be sufficiently specific and quantitative, even more than in the case of systems due to the increased level of parallel autonomous developments. In this case study, we illustrate the use of scenarios for that purpose. Quantification and scenarios can help to manage the distance between model and reality by calibrating and validating the numbers.

Existing Practices: Distribution Centers and the Logistics World

In this section we introduce the domain of distribution centers (DCs) and their role in logistics of goods. DCs facilitate aggregation and repackaging of goods for later transportation.

Figure [Logistic Chain] shows a simplified logistics chain with its goods and information flows. A wide variety of cargo transportation means, ranging from trucks to planes, are used to transport goods from suppliers to DCs. Similarly these means of transportation are used to transport the goods to their destinations. The type of transportation used is adapted to specific

circumstances; e.g., in inner cities small vans will be used. A complex network of information systems controls and monitors the goods flow. Sales and purchase departments place orders for products, based on actual sales and planning forecasts. Administrative and financial departments manage the related money flows.

We depict Figure [Logistic Chain] as a single chain while, in reality, the whole process might consist of multiple producer-consumer steps that are co-mingled within intertwined logistics chains. The focus of this case study is on the DC, a complex SoS. We need an understanding of the key drivers in the logistics chain to fully understand the trade-offs of the specification and design of DCs. The simplification of Figure [Logistic Chain] allows us to discuss most logistics key drivers without too much loss of realism.

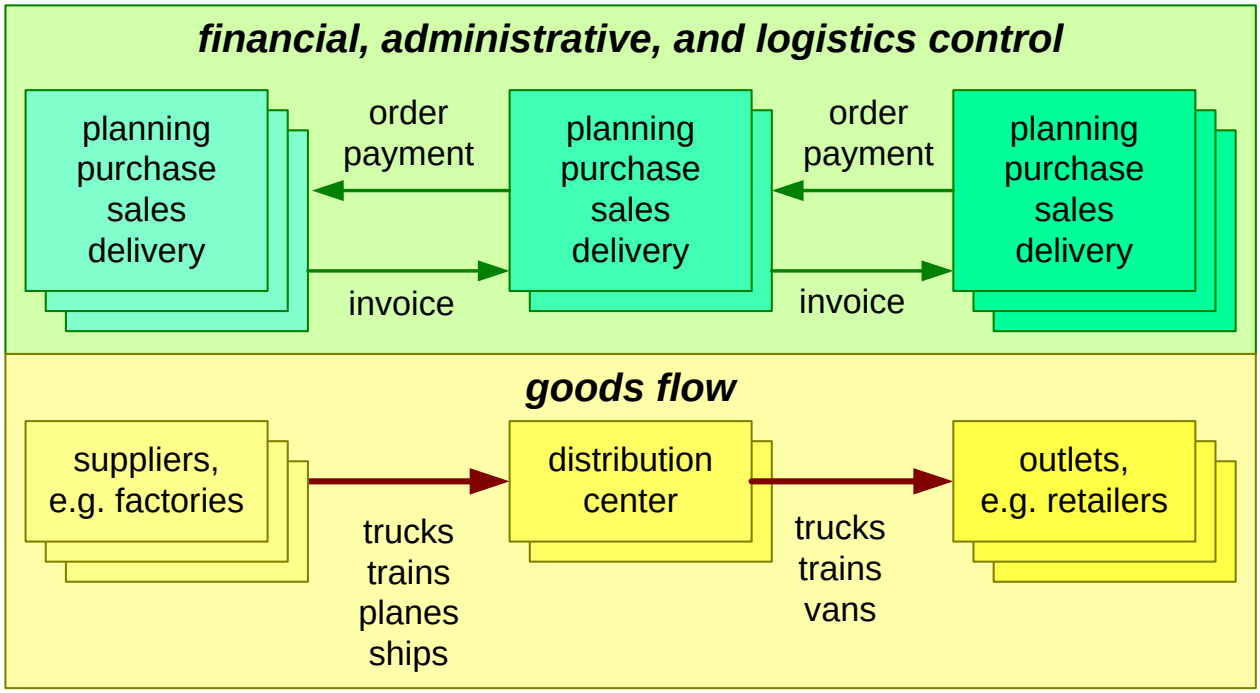

Figure [Logistic Chain]. The logistics chain consists of goods and information flows. These flows are strongly coupled. Return flows of goods are not shown for sake of simplicity.

Characterizations: Project and Process of Distribution Center Business

The DC design and realization business is a typical project business. The DC suppliers and their customers (logistic service providers, who operate DCs) discuss specifications and conditions during a tender phase. The customer selects a DC system supplier based on the offers from multiple suppliers. Once the customer places an order, the supplier starts a delivery project. After installation and acceptance, the customer pays for the main part of the contract. DCs have an operational lifetime of many years to even decades. During the lifetime, upgrades can occur via a similar tendering and delivery process. Figure [Project Life Cycle] shows a typical project life cycle.

Business models are evolving in this business. In the past projects usually finished after delivery. An increasing amount of business is generated through services during DC operation, where service level agreements define performance and payment levels.

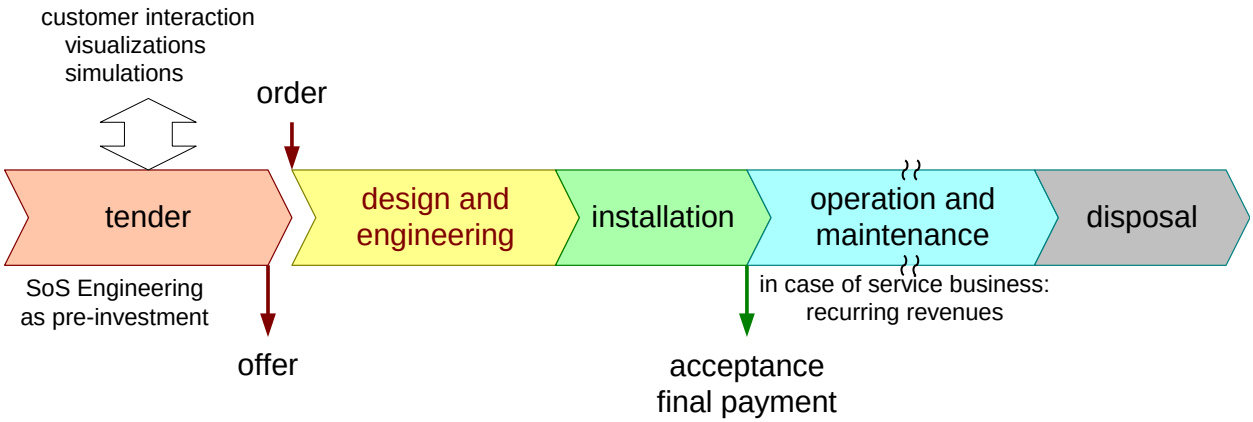

Figure [Project Life Cycle]. Project life cycle for DCs.

The tendering phase is for the suppliers an initial investment where senior engineers start the systems engineering activity. In this business, suppliers use visualizations and simulations to

communicate with customers about the operations in the DC, and to estimate performance in terms of throughput and meeting shipment deadlines. These simulations, together with the high-level design of the quoted system, are the starting points for engineering once customer and supplier sign the contract.

Guiding Principles

Central to the case study is the tender phase as sketched in Figure [Project Life Cycle]. The pre-investments that are done by systems engineering (SoS engineering) before actually selling the DC system play a critical role in the system's business success. Insight in the system-level characteristics is crucial for a sound balance between trustworthy system concepts on the one hand that are not over-dimensioned on the other hand. In our opinion, the manageable and communicable connection between (model-based) understanding at different levels of SoS aggregation is a well-suited means to achieve this goal.

Design of Distribution Centers

Various types of DCs support a wide range of different supply chains with different business processes. Business process variations include distribution of order sizes and their frequencies, and the range of different goods to handle such as retail, frozen food, fresh food, clothing, jewelry, and spare parts.

To design efficient and effective DC shop-floor processes, i.e. the internal DC's daily operations, DC designers have a large range of different design patterns at their disposal. These design patterns are used to compose a complete system from a large range of standard material handling components and engineered specials where necessary. A flexible control system finally integrates all components and controls the shop-floor processes.

In addition to designing the material handling system, designers also have to decide how human operators (performing material handling functions) do fit as part of the system, i.e. designing the user interfaces of the system that facilitate performing the manual tasks.

Figure [Artist impression] gives an artist impression of an entire DC. The inward and outward bound shipping docks are at the left hand side, where the trucks are sketched. The large area with high scaffolds is the bulk storage area. The low areas at the right hand side are the pick and place workstations.

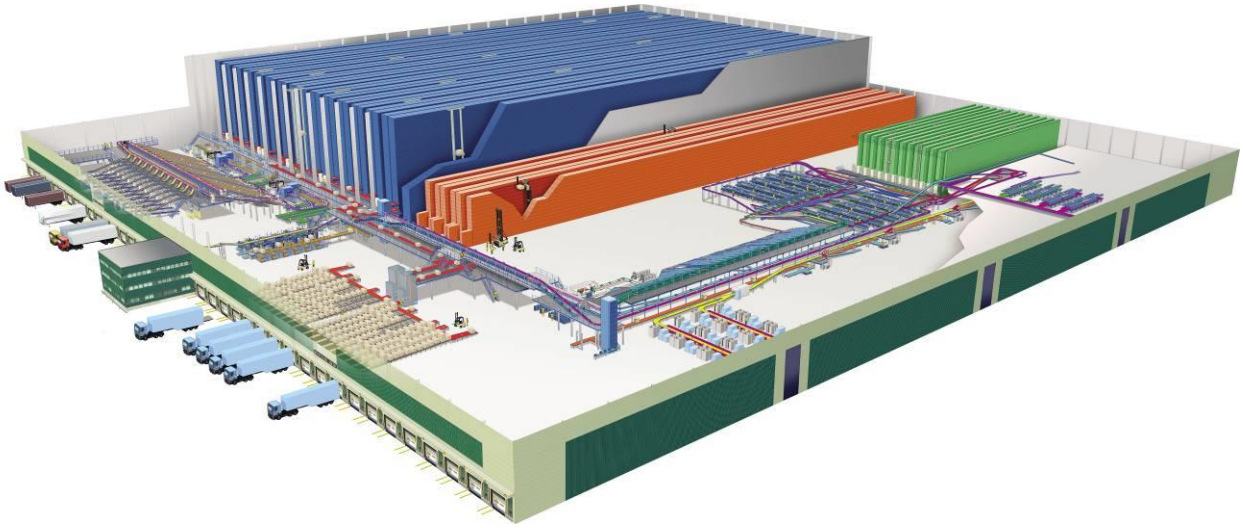

Figure [Artist impression] Artist impression of an entire DC. The blue area is the main pallet store, transport, and pick and place, the green area is a tote store for pick and place. The orange area is storage of oddly sized goods that are handled differently. Courtesy of Vanderlande Industries B.V.

Although DCs have many different types and different shop-floor processes, their global functional view is quite stable. In Figure [Functional Model DC] we sketch such a functional view that already hints towards different implementations for different flows.

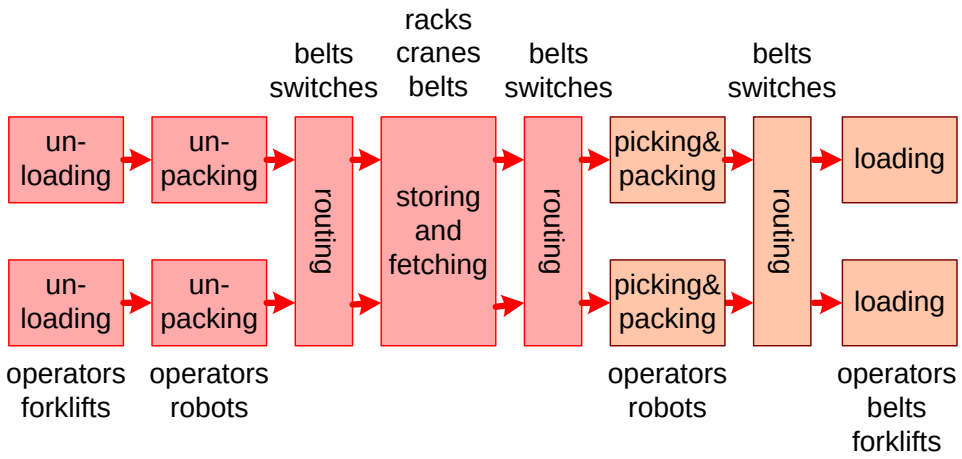

Figure [Functional Model DC] Generic functional block diagram of a DC.

Purpose

History

Conventional DC suppliers have their origin in other types of businesses, which are mostly related to one of the constituent systems of the current DC systems. This means that the responsibilities for the operations at DC system level are relatively new to most DC suppliers; in earlier days, these were residing with the owner of the warehouse. As the need for more effective and efficient operations became stronger over time, several companies seized the opportunity and expanded their business from being single system supplier towards integrating system-of-systems supplier with a servicing business on top of it. This is where systems engineering of the larger scoped SoS came into play.

Current Situation

Nowadays, DC suppliers offer complete DC systems to their customers. The role of system engineers is to generate and validate system concepts that fulfill the customer needs within the

constraints of a valid business case for both parties. Generation and validation is done by experience and performing fairly detailed simulation experiments with a limited set of options. DC suppliers use these simulations to show the customer how the proposed system will actually operate. Different user-scenarios are preferably based on real user data and might represent contractual value, i.e., play a role in the customer acceptance test of the delivered system.

Known Problems

Goods flow characteristics at different levels

The whole logistics supply chain is highly dynamic. Sources and destinations may change and varying quantities and types of goods are flowing, which are seasonal or fashion-dependent, for example. As a result, logistic service providers continuously extend and adapt their DCs. These companies will also make trade-offs between suppliers and DC locations (distance to outlets) and order sizes and frequencies. The size of the packaging units in its turn will depend on order size and frequency as well.

The logistics supply chain and the dynamic role of DCs

Many stakeholders with varying concerns and interests are present in the chain. Incentives for making changes to streamline the logistics flow are not always in place. For example, packaging has a huge impact on the goods handling during transportation and in the DC. However, the stakeholder responsible for cost of packaging might have no interest in simplified product handling in the DC.

Mission and Desired or Expected Capabilities

The placement of products on a pallet or in a tote is a process with a wide variety of stakeholder needs. The recipients of the products want undamaged products, e.g., heavy products stacked on

top should not crush fragile products. Transporters need a stable stack on a pallet or tote that will not deform or shift during handling, and they need minimum use of space, i.e. pallets and totes packed as full as possible, because transporting "air" is very expensive. Retailers and drivers may prefer a stacking order of products that fits the sequence of delivery. Human operators place products with knowledge of these stakeholder needs.

System of Systems

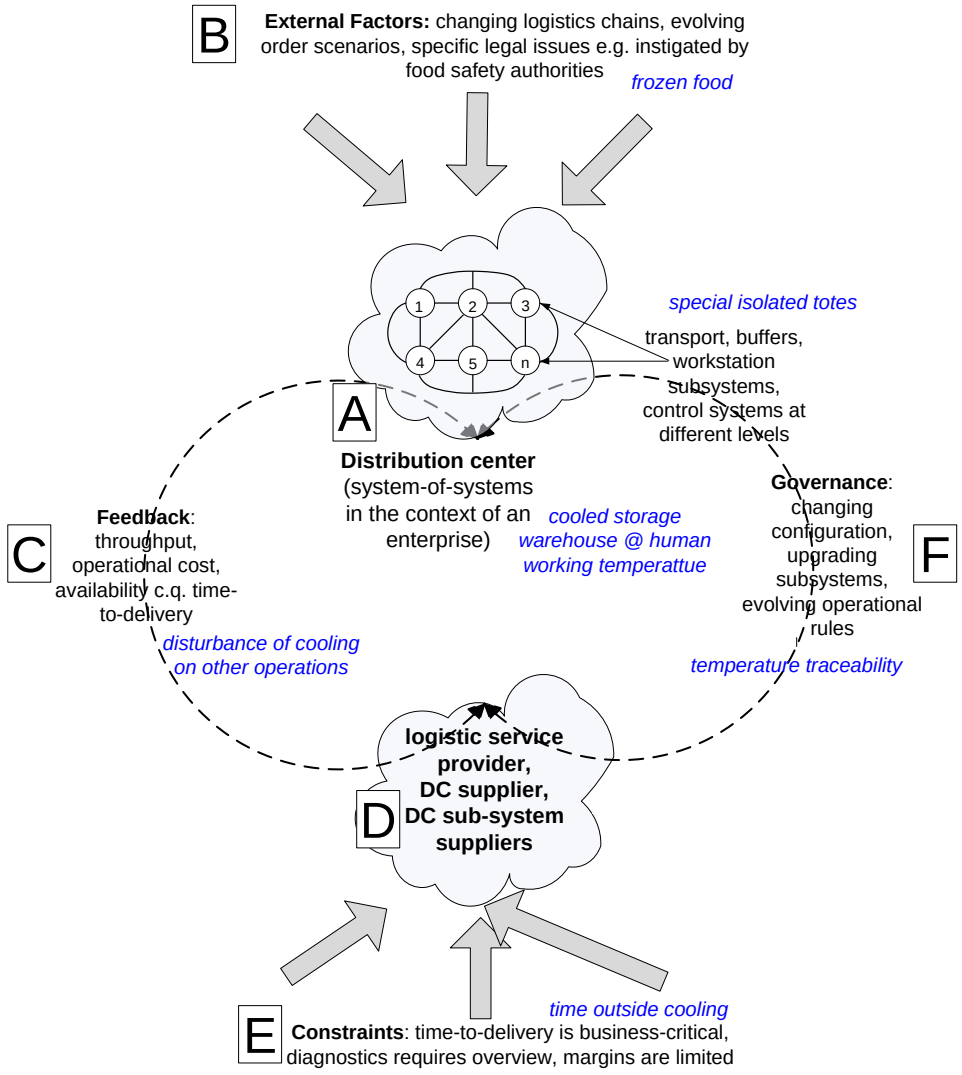

Figure [SoS] An overview of the distribution center as a system-of-system. In Italics a specific example is shown

of the impact of adding the handling frozen food to an existing warehouse.

The context of a distribution center as a system-of-systems is shown in Figure [SoS]. The DC itself (A) is contained in the larger context of enterprises. In this chapter, we have mainly focused on the conception phase of the DC, in which case the governance C, feedback F, Constraints E, and external factors B in Figure [SoS] guide the what-if scenarios that have to be considered by the system engineers. In this phase, the governance body D is mainly the system engineer. In an operational setting, a clear governance body is often absent or spread over diverse parties.

As example, we can look at the impact of adding the handling of frozen food, an external need (B), on an existing distribution center (A). This requires the addition of a cooled storage area for the frozen food. The supplier packs the frozen food in special totes that isolates the food from the warmer external environment. There are regulations (F) for the traceability of the food temperature. If the food in the isolated totes is too long in warmer environments, then the food temperature may rise too much. The suppliers (D) will all ensure that the totes are not too long exposed to higher temperatures. The special totes and cooled storage area imposes constraints on the operation. Suppliers will communicate (C) impact of the addition of frozen food handling to the distribution center (A).

Environment

Different logistic service providers have different strategies in choosing locations for their DCs: de-centralized, centralized, or a hybrid approach. The location strategy depends on the types of goods and business, and involves a trade-off between warehousing and transport in the logistics chain. The end customer ultimately pays the cost of transport and warehousing for products. Pricing strategies and business models determine how cost and margin are distributed over the

chain. This distribution may change over time.

Scope

The scope of the system for this case is the DC itself. Considerations about the logistics chain and enterprises around the DC are inputs to the specification and design.

Structure

The main functions of a DC are to receive, store, and redistribute goods, the last step probably in other combinations of goods than the first step. In order to fulfill this functionality, it is relevant to look into the basic packaging concepts. The left-hand side of Figure [packaging concepts] shows how an operator composed an order consisting of multiple cases onto a single pallet. A truck transports the pallet to its destination. The right-hand side of Figure [packing concepts] shows an alternate approach to packaging. The DC designer introduced a standard box, a so-called *tote*. All DC functions can use totes as standardized means to transport, handle, and store items.

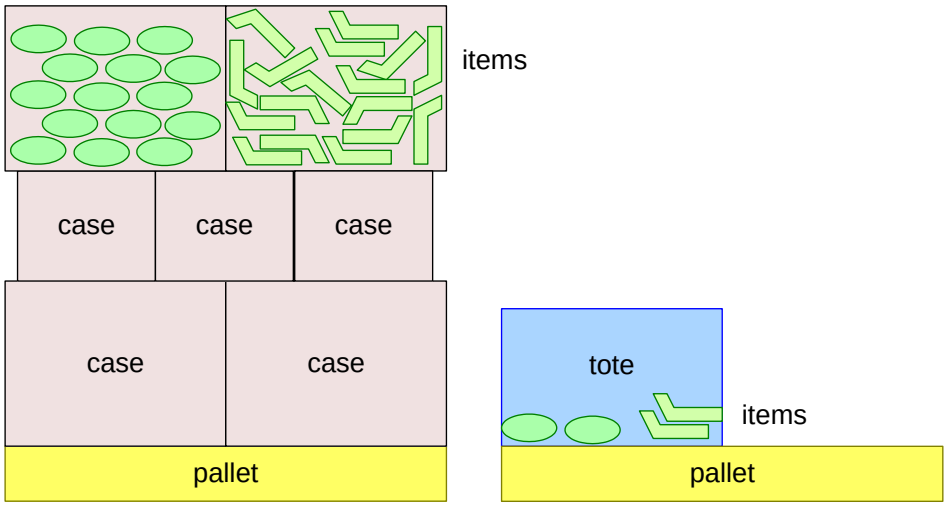

Figure [packaging concepts] Packaging options of outgoing orders. Left-hand side is case based (cases delivered by suppliers), right-hand side is tote based (tote composed at the DC).

Boundaries

The packaging concepts of Figure [packaging concepts] play an important role at the boundaries of the DC. Next to the clear use of pallets outside the DC, the use of totes goes beyond DC boundaries as well; they may be transported to and from retailers, and in a few cases are used by suppliers. This decreases the amount of superfluous goods handlings as well as to increase the quality of transport.

Next to the physical part of the boundaries, there are information interfaces for order input, tracking, goods traceability, etc. Typically, this is a transactional interface at the ERP level. For DC suppliers this is mainly limited to an interface to the ERP system that is owned by the logistic service provider.

Internal Relationships: Cross-docking, Case and Item Picking to Manage Goods Flow

Retailers place orders at the DC with a well-defined number per product. In the DC, operators compose the order in the so-called picking process. The rudimentary form is cross-docking: orders consist of complete pallets with goods. In cross-docking, the DC is used as temporary, e.g., 1 day, storage of pallets that are retrieved on basis of a customer order. The next form is case picking: orders consist of cases with cases containing multiple items. Picking is easier when it just involves cases, since cases are easier to handle. However, case picking constrains order flexibility. Item picking is the third approach that can fulfill orders exactly, because the picking is done on the smallest defined unit.

In the last two approaches, operators fetch the proper number of cases or items from the stock and place it in an order tote or on an order pallet. A transportation and routing system brings cases or totes with one type of item from the storage to a pick-and-place workstation, see Figure

[Pick and Place], and returns remaining stock to storage if needed.

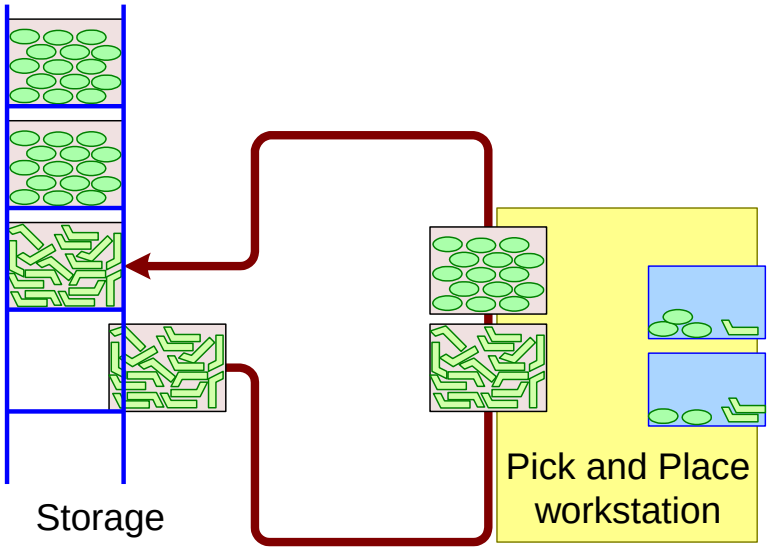

Figure [Pick and Place] A typical pick and place flow of goods.

Figure [Photo Pick and Place] shows an actual pick-and-place workstation. The six cases left and right each contain items for one order; the tote in the middle contains the stock. This increases operator productivity: For fast-moving products the stock in one product tote can fulfill the demand for multiple orders in the workstation. The central monitor shows the total quantity to pick from the product tote, and six small displays on the sides show the quantities to place in the order totes.

Figure [Photo Pick and Place] An item pick-and-place workstation (PICK@EASE.4 from Vanderlande Industries B.V., see http://www.vanderlande.com/). Totes at the left and the right with the small displays on top contain orders; the tote in the middle with the large display contains the stock items.

External Factors: Some basic concepts of Goods Flows

Suppliers package quantities of homogeneous goods (items) to deliver them to DCs. Appropriate packaging (i.e. efficient, secure, and safe) facilitates transportation and handling. The destinations of the goods, e.g., retail shops, need a mix of various goods, probably from different suppliers. Dependent on the quantities required by the outlet channel (the retailer), different packaging units can be delivered. The information systems that control the DC send the required goods to a workstation, where the operator picks the right number and composes them into an order.

Constraints

Items may have all kinds of shapes, might be fragile, slippery, or the opposite: rough. Cases

provide some protection for fragile items, and provide some size and shape standardization, but are not always optimized with respect to the logistics chain's requirements. Suppliers typically determine the case size and material, probably based on requirements from the retailers and cost considerations.

Sponsor of Governing Body in Concept Phase

In most cases for this type of SoSs, the governing body is ill defined in its operational phase: it is spread over a multitude of parties, including the DC supplier that also defines the SoS. This situation is taken into account by the system engineers that create the SoS concept and validate it against different (change) scenarios. These system engineers form the *de facto* governing body in the SoS's conceptual phase. This role and the related activities that we describe in this chapter are sponsored by the DC supplier organization at senior management level due to the visibility the DC supplier has in the resulting SoS once it has become operational.

Challenges

Starting from the domain characteristics as sketched in the background section, we will highlight a number of challenges that DC suppliers and the involved systems engineers face. These challenges are further aggravated by the need to include technological developments to address changing customer needs.

DC suppliers: challenges and new technology development

DC customers span a wide variety of needs, in size, capacity, capital cost, operational cost, types of products, rate of change, timing needs, etc. The result is a lot of diversity in composition of DCs. A core challenge is to determine whether a specific system configuration works and is

economical. Currently, systems engineers of DC suppliers mainly address this challenge by experience. A complication is a continuous shift of boundaries of the system, e.g., suppliers of systems as one-time deliverers of components, such as conveyors, up to service providers using service level agreements. Many suppliers of DCs have a history in systems like conveyors and evolve into system providers, and later into solution providers.

The key drivers of DC operation are the following:

- Throughput
- Time to delivery
- Operational cost
- Capital expenditure

These key drivers form a field of tensions. For example, many possible measures to increase throughput lengthen the time to delivery, especially in the presence of (physical) errors. Similarly, decrease of operational cost often requires an increase of capital. Personnel cost dominates operational cost in the current state of DCs in Europe and North America. The pick-and-place process is most costly, followed by unpacking. Lack of personnel availability, certainly in Western countries, threatens throughput and time to delivery.

DC providers are therefore looking into increased automation. Advances in robotics technology in combination with its decreasing cost may facilitate further automation. Main challenges for the economic use of automation are coping with:

- A variety of products, e.g., in size, shape, rigidity, fragility, slipperiness, etc.
- Continuous changes in products and product mix
- Continuous changes in logistics process, product handling, and transportation

The SoS perspective on this transition is that in the current situation human operators fulfill a

crucial role in making processes work with existing systems. Operators recognize fragile products and know to handle such products carefully. These human operators adapt their workflow to cope with such constraints. E.g., an operator might park fragile products at the side temporarily, so that they can be stacked on top later. What will happen when systems get more tightly integrated? What will happen, when no or less human cognition and intelligence is available to cope with unforeseen issues? For certain, the systems will have to become much more intrinsically robust, at each individual level as well as across different levels of operation. To provide an example for this in the area of the picking functions, Figure [Human Vs Robot] shows that replacement of humans by robots trigger paradigm shifts for many sub-functions:

- Intelligence and adaptability of the human brain are replaced by analysis and control functions in software.
- Eyes and human senses are replaced by cameras and sensors.
- Arms and hands with many degrees of freedom (DOF) and fine control are replaced by robot arms (or other mechanical topologies, such as x, y, z oriented actuators).
- Legs for transportation may be replaced by increased range of the robot, or by any kind of transportation system; pick and place can be done at completely different places in the DC.

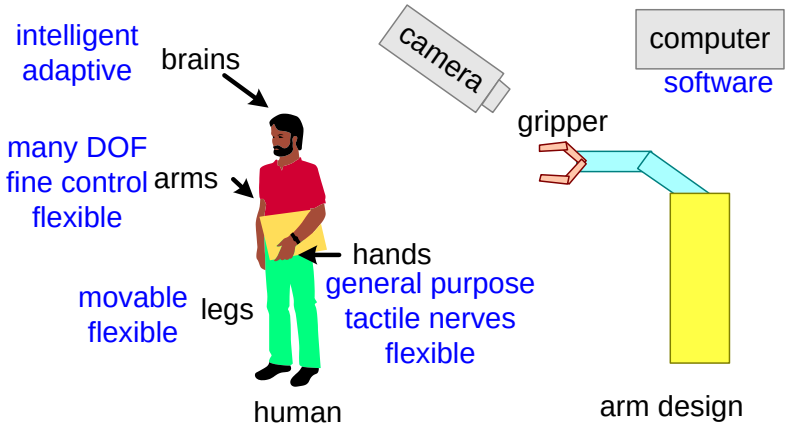

Figure [Human Vs Robot] Replacing intelligent and flexible humans by robots is a paradigm change for many sub-functions.

Most changes create new challenges mostly related to intelligence, adaptability, and flexibility. However, other benefits may offset the challenges, e.g., indefatigable, strong, fast, large range, and robust against noise, temperature (e.g., handling frozen goods), reduced lighting and heating to accommodate human operators, avoiding limitation of labor regulations, and many other environmental aspects. DCs can potentially operate continuously with lower operational cost by replacing humans by robots.

Systems engineering challenges

Systems of systems (SoSs) comprising autonomously developed systems have a number of particular challenges for systems engineers. The independence of individual systems complicates systems integration. Will the aggregation of individual systems behave and perform as intended? The high-level of complexity of an SoS limits the overview that designers have of the system, and may threaten their ability to reason about individual systems and the SoS as a whole. Systems engineering of a DC suffers from the well-recognized systems engineering problem of

many components with various degrees of freedom and levels of uncertainty. The mapping of business processes to actual components proceeds along various levels of abstraction. An example of these levels in the case of DCs is given in Figure [SoS levels], where the enterprise level is connected through the warehouse SoS to specific systems, such as the pick and place station to be automated, which consists of several subsystems and their constituent components. Characteristic for the SoS situation is that more abstraction levels are present and that part of the levels and systems are outside the scope of control and may even be undefined.

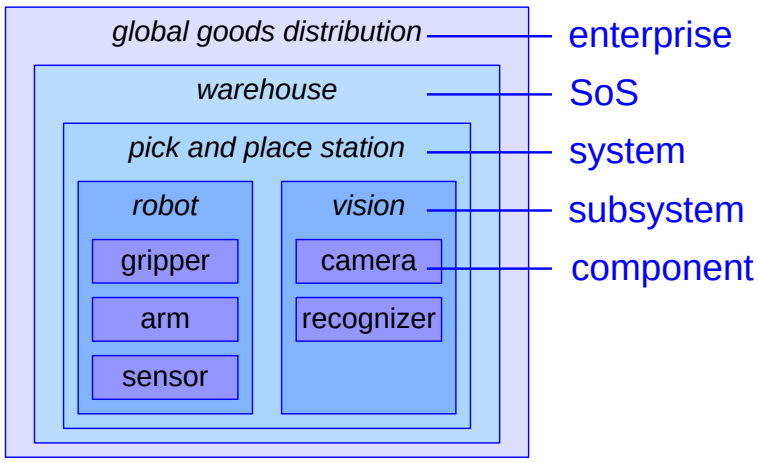

Figure [SoS levels] Summary of "system" levels in this case from components such as grippers and cameras to an enterprise for global goods distribution.

Learning by extensively testing system candidates is and will remain too expensive. How are we to guide the systems engineering process quantitatively in a cost-effective way? How can we get input data, and how can we validate concept choices? We present two related methods here. The first, exploration by scenarios, is well suited to scope the problem at a sufficiently high level of the system, while at the same time it indicates how the challenges translate to lower levels. The second method, which encompasses modeling, simulation, and analysis, is illustrated by a

number of interrelated examples, which connect the scenario-based analyses at the SoS level to the levels below in a more quantitative way.

Development

Systems Engineering: Exploration by Scenarios

Systems engineering promotes the use of stories and scenarios to problem and solution spaces. We will discuss a limited number of scenarios with increasing degrees of difficulty to explore the challenges formulated before, and to see how different concepts can contribute to fulfillment of the key drivers.

Scenario 1: high volume drugstore chain

Scenario 1 comprises a chain of drugstores where a high volume of products is flowing through the DC. Individual drugstores receive cases with products, so individual item picking is not needed. Figure [High volume drugstore] shows the scenario at the top. This particular case can be served by a robot with x, y, and z as movement directions, and a gripper with only one degree of freedom. Placing cases with a limited range of sizes and weights on pallets is still challenging to do in a completely automated way. The main success factor for this scenario is effective planning and control across the logistics chain.

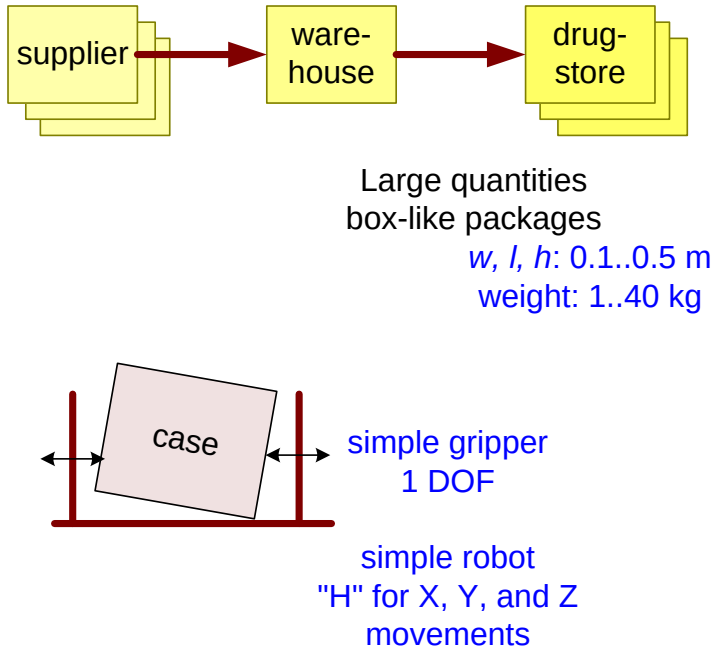

Figure [High volume drugstore] Scenario 1: high volume drugstore. The DC distributes medium size cases with a limited weight range in high quantities.

Scenario 2: High dynamics extended drugstore

Some drugstore chains have extended their business (Scenario 2 of Figure [High dynamics]) by selling all kinds of additional products. These products are often one-time specials, season (e.g., Christmas), event (e.g., Olympic games), or fashion related (e.g., sunglasses). This extension has several consequences:

- There is more variation and dynamics in suppliers; suppliers may appear and disappear in a short period.
- There is more variation in the size and weight of items and cases

Humans can still handle sizes and weights. However, the basic mechanical system of the high-volume drugstore is inadequate. The sizes vary so much that probably multiple grippers are needed to cope with size differences. Shapes might be less uniform as well, complicating

grippers even more.

A main characteristic of such a DC is its continuous operation, where there is no time or competence to adapt parts/systems to cope with some new variety. Solutions where a robot or gripper has to be programmed or taught how to handle packages do not fit. Similarly, introduction or change of suppliers should happen seamlessly and instantaneously.

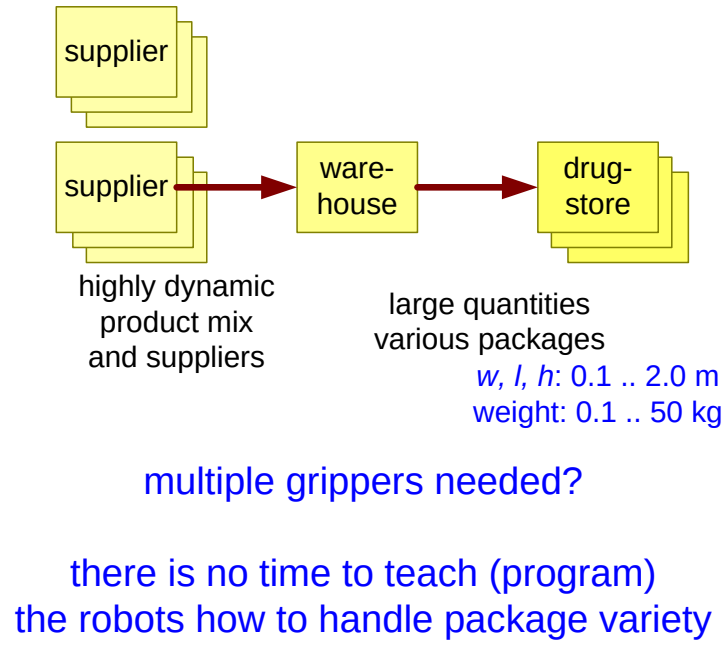

Figure [High dynamics] Scenario 2: A chain of drugstores that has extended its business causing much more dynamics in the product mix and in product sizes.

Scenario 3: mail order company

Another extreme scenario is a mail order company, where orders typically are one or only a few items. An additional complication is that customers frequently return orders that have to be processed and if possible restored in the DC. The variation and dynamics in products is similar to Scenario 2, the dynamic extended drugstore. The consequence is that the DC now needs item picking as functionality, which increases the requirements for the gripper further. We cannot longer assume minimal case sizes or shapes, and we may have to cope with fragile items. Surface

texture of items might be an issue; what can the gripper grasp? What can the vision system reliably recognize?

The limitations of grippers and vision systems could probably be mitigated by packaging of individual items, i.e. by other systems in the chain. However, packaging serves other purposes too, such as branding, and additional packaging means additional cost and processing steps. How can we achieve a globally improved chain, with a distributed ownership of the systems within an SoS?

Distribution of fresh food, see Figure [Fresh food distribution], introduces new needs, such as the capability to cope with fragile products (eggs, tomatoes), packaging constraints (such as transparency, waterproof, airtight), strict ordering rules for expiration dates, and fast integral delivery time. Fragile products and packaging constraints again affect concept choices for automated pick and place. For instance, the vision system becomes more complicated; can it work sufficiently reliable and fast?

Scenario 4: fresh food distribution

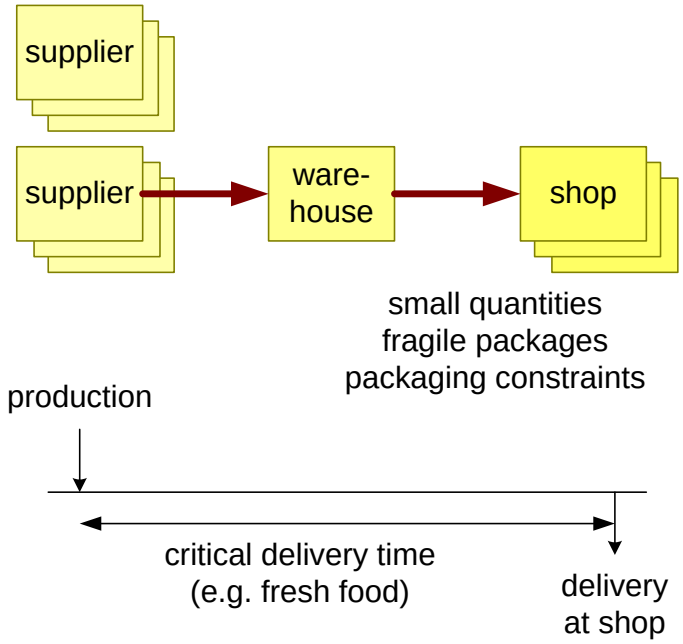

Figure [Fresh food distribution] Distribution of fresh food, where the food may be fragile, packaging has to meet special constraints, and where delivery time is critical.

The integral delivery time depends on all functions in the chain from harvesting or production to final positioning for display. Concepts that may improve DC capacity, such as large batch sizes and long belts serving many pick-and-place stations, may increase delivery time.

Management of Uncertainty

Systems engineers use scenarios to make uncertainties explicit and tangible. Within any given scenario further uncertainties are present: how will the business develop in the future, both in terms of customers and the orders that they will issue, how strongly will the assortment change, what are expected properties of goods that have to be handled, etc. Not all of these uncertainties are considered independently: in order to manage them they are limited by considering certain combinations. On the one hand this leads to a classical risk management approach (what are

chances that certain combinations will occur, and what would be their impact), while on the other hand historical DC data is mined to obtain a realistic view of variations that will occur over time.

Systems Engineering: Modeling, Simulation, and Analysis

In this case study, we argue that systems engineers should not use what seems like unmanageable system complexity as an excuse to set their analytical capabilities aside. We propose the use of multi-level models to reason about individual systems, critical design choices in systems, and the expected performance and behavior of aggregated SoSs.

The design of DCs can be characterized by a large degree of design options, for a wide variety of customers. The challenge for the DC supplier is to design sufficiently trustworthy in the tender phase (see Figure [Project life cycle]) to be able to make a feasible offer (performance, cost, time) that still meets overall business goals. Suppliers typically build simulation models to analyze design options, and to communicate with customers. Simulators thus serve a dual purpose:

- Facilitating design and mitigating project risks, by validating the predicted performance characteristics of the system that is being proposed.
- Communicating with customers and convincing them of the value proposition in the offer, by visualizing the system and showing them the behavior and performance of the system.

The consequence of this dual purpose is that suppliers tend to make simulators at a moderate level of detail. The underlying models are dynamic and executable, with inputs mimicking the real world. Nice visual front-ends ensure engagement of customers.

What tool or model at what level of detail?

Any system, such as a pick-and-place workstation can be viewed at various levels of detail. Figure [System pyramid] (text and figures in this section are based on [Muller 2011, Sections 2.4 and 4.1]) visualizes this as a pyramid with the vertical axis an exponential scale for the number of details. The most abstract way to look at a system is to see it as, for instance, "a pick-and-place workstation" (at position 10^0). The system can be specified in its ~10 key performance parameters and functions (position 10^1). Elaborating these key performance parameters further results in a system specification (somewhere between 10^2 and 10^3). The bottom of the pyramid represents all details produced by engineering that define the system: mechanical properties in CAD-M files, electrical data in CAD-E and software source code. The bottom of the pyramid depends on the type of system; many systems nowadays have over 10 million details. These details are typically defined, stored, and maintained per discipline. The detailed designs tend to be less extensive (position 10^6). From system specification to detailed design, a multi-disciplinary design takes place: partitioning in subsystems, components etc., functional design, function allocation, interface definitions, etc.

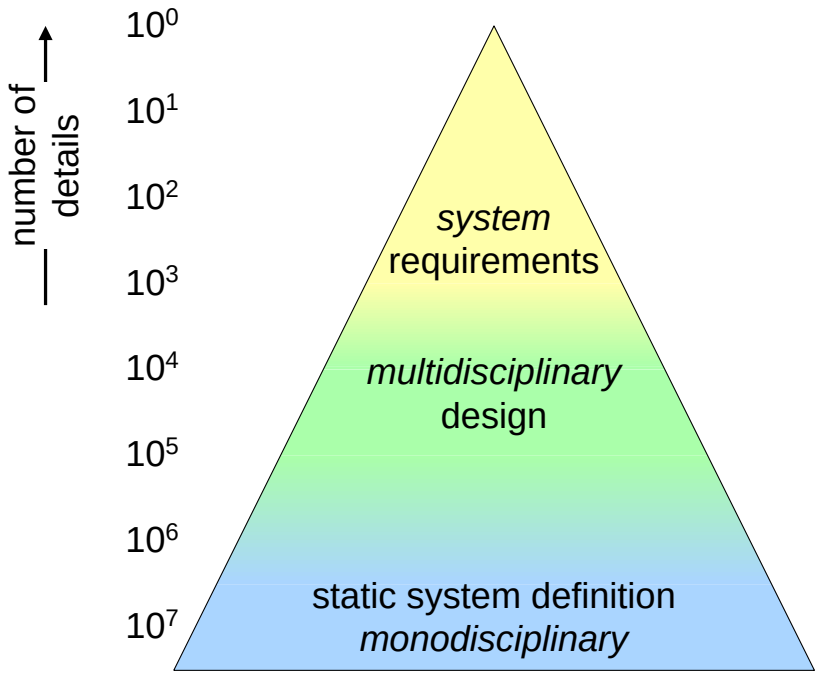

Figure [System pyramid] Various perspectives at a single system, from highly abstracted ("the pick-and-place workstation") to all static engineering details as stored in engineering databases.

The DC supplier delivers many systems, hence the pyramid reflecting all delivered systems will be significantly larger and reach the order of magnitude of 1 billion details. However, due to its SoS nature, there is a lot of detail outside the delivered system that is relevant for the specification and design of a DC. Figure [Diabolo Tools and Models] visualizes this by adding an upside-down pyramid with a similar scale for the number of details. As systems engineers we simplify the outside world to its main stakeholders and interfacing systems. However, we can model the context in more detail, using business process models, and chain and enterprise models. However, the real context is full of unique, individual people, suppliers, retailers, and goods, all with their own characteristics. The context can easily exceed one billion details as well.

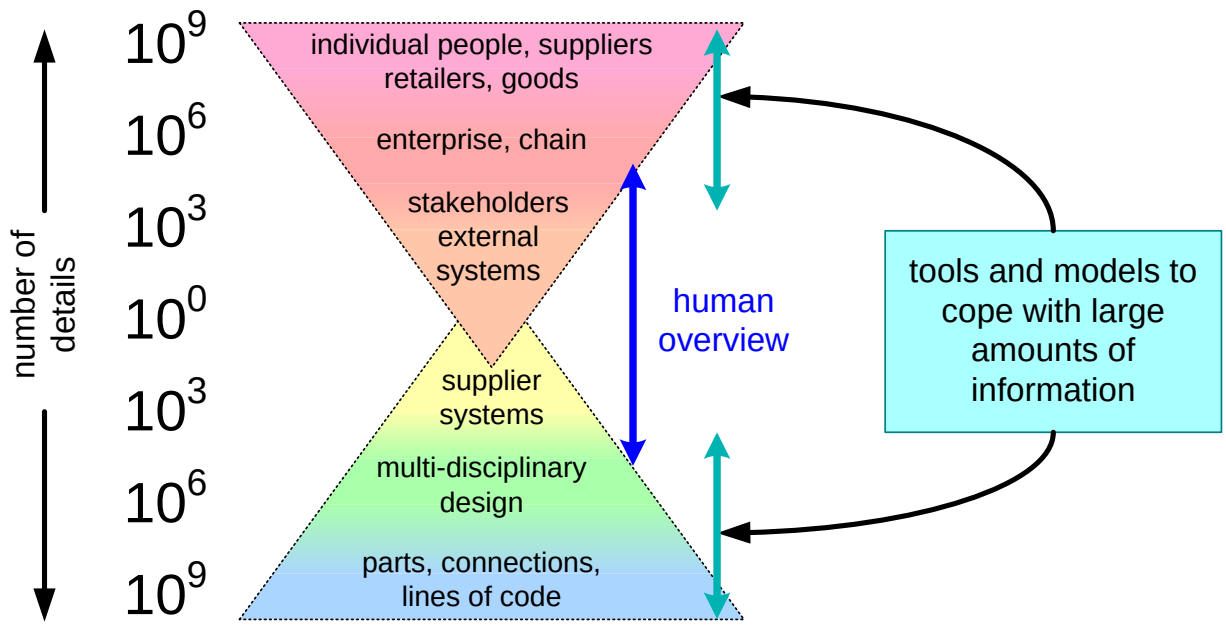

Figure [Diabolo Tools and Models] More perspectives for the systems delivered by the supplier at the bottom, and the context of the supplied systems at the top.

Figure [Diabolo Tools and Models] illustrates that human reasoning and communication typically is based on abstractions. System developers use computer assistance to manage all engineering details. Human reasoning is based on a rich pallet of representations and models, e.g., physical and functional, graphs, and formulas.

In Figure [Diabolo Models] we have positioned the simulators that are typically used in the tender phase and afterwards for DC projects. These simulators contain a moderate level of detail, such that customers and designers perceive the simulator outcome as realistic. The level of detail of these simulators also has some disadvantages:

- Effort of creating, verifying, using, and maintaining is roughly proportional to the level of detail.

- The cycle time to explore concepts is also roughly proportional to the level of detail.

Architects of the models strive for partitioning of the models and standardization of information and interfaces to minimize the effort to use and adapt models. Well-architected models suffer less from these disadvantages then ad-hoc models. However, even well architected models with lots of detail can slow down exploration. The effort increases especially when the exploration goes outside the foreseen exploration space, e.g., for new concepts.

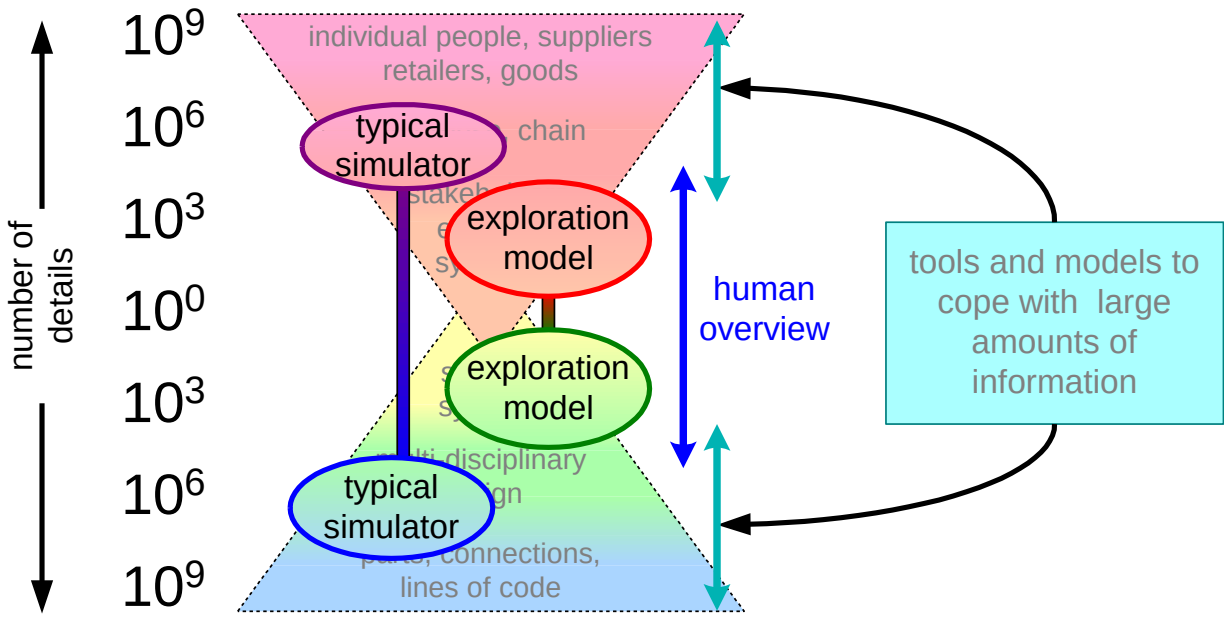

Figure [Diabolo Models] Positioning of simulators and exploration models.

Results

Examples of Models

In this section, we will discuss multiple models at various levels of abstraction. The purpose is to show the relation between the objectives of an individual model and its level of abstraction. Most of the models are used in combination with other models. For example, modelers can integrate a

crane performance model into a DC performance model, or they can use the crane performance model to parameterize the DC level model properly. Similarly, models of production and order flows can feed a DC performance model.

For most models, we provide an estimate of size and effort to make this particular model. Often, the creation of the model reuses code from similar activities. For simple models that can be tens or hundreds lines of code. Larger simulations, may build on tens of thousands reused lines of code. Reused code facilitates fast creation of models. However, it may slow down developments when the reused code does not fit the problem. Most modeling effort relies on huge amounts of "black-box" tools and toolboxes, e.g., Matlab, Eclipse, or other simulation environments.

One of the concepts proposed for a customer similar to Scenario 1 is to simplify the packing by delivering all cases in a fixed pre-defined sequence. For this customer the stacking of the cases is critical. Figure [Model Fixed Sequence Picking] shows how the picking function is realized by fetching from the storage, the sequencing function by inserting cases in the right place in the transport stream and the packing is performed at the end by an in-sequence palletizer.

ACP (Automated Case Picking) model to analyze effect of fixed sequencing on performance

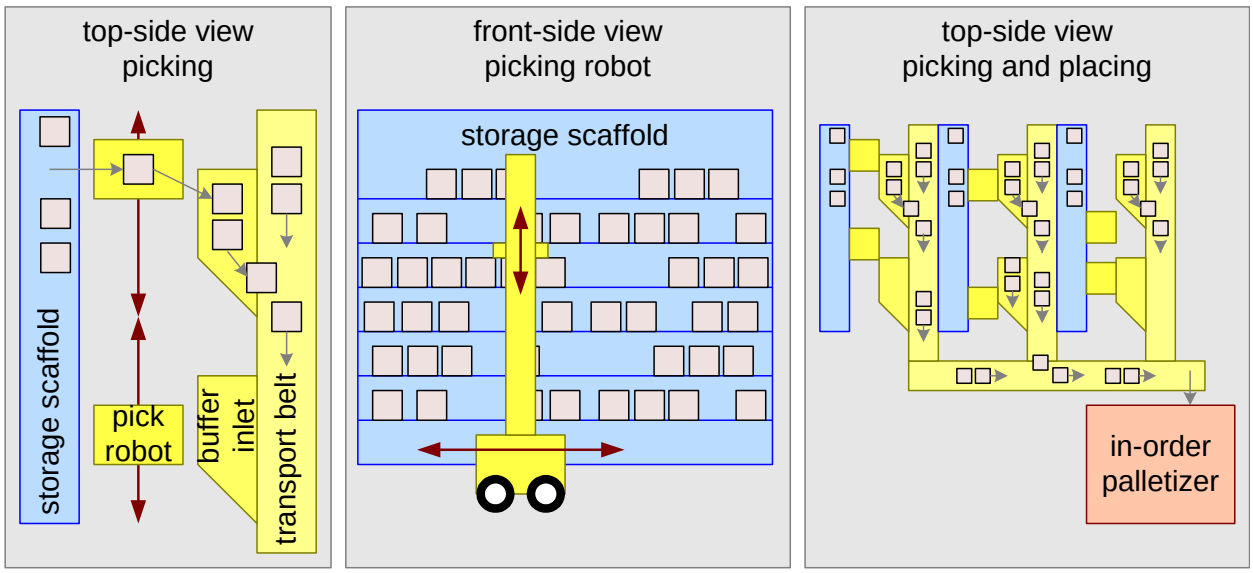

Figure [Model Fixed Sequence Picking] Model to analyze the effect of fixed sequencing in case picking on performance.

At several places in the goods flow we need buffers to decouple operations; e.g., between fetching goods from scaffolds and inserting goods in the transport stream. Buffers are typically First-In-First-Out (FIFO) buffers, since these are much simpler and lower cost than random access buffers, especially for buffers dealing with physical goods. Random access buffers would have provided more freedom in scheduling the operations in the goods flow.

The Falcon researchers made a model to see the impact of the in-sequence requirement of the palletizer at the end on the performance for different transport configurations of the SoS without random access buffers. This model is a highly abstracted dynamic model. It only takes the picking, merging, and insertion into account to study impact of various topologies. Figure [Sequencing Topologies] shows a number of example transport topologies to indicate the type of variations that were studied with the performance model. The model simplifications facilitate a comparison of concepts without achieving absolute performance numbers. That is, the absolute performance figures will exhibit large accuracy spreads due to many neglected influences, but the relative performances for different concepts are significant due to sufficient precision in the model analyses.

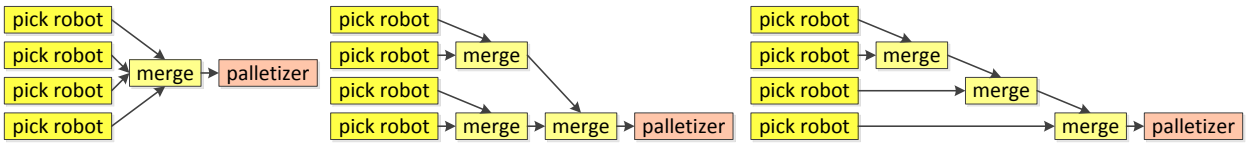

Figure [Sequencing Topologies] Examples of transport topologies with FIFO buffers at every merge step. These configurations served as inputs to the performance model sketched in Fig. [Model Fixed Sequence Picking].

In retrospect one simplification, expressing intermediate buffer sizes (how many cases can be temporarily stored before being processed) in number of pallet tasks instead of cases, turned out to be counterproductive in terms of communicating about the model and its results.

The model made very clear how large the effect of intermediate buffering is on throughput. A static model of the penalty of small buffers was derived showing that the penalty is roughly proportional to $1/\sqrt{B}$, where B is the total buffer space in the system, evenly spread over the merging sub-systems. However, cost, space, and probably cycle time are proportional with B, showing trade-offs to be made.

This model is described in more detail in Chapter 6 of [Hamberg 2012] and in [Roode 2011].

The model contains less than 200 lines of code. Construction of the model and performing analyses involved three days of work. Such models have influenced the choice for configurations of Automated Case Picking (ACP) systems.

Crane performance model

This model zooms in on a picking robot as shown in the middle of Figure [Model Fixed Sequence Picking]. The picking robot is realized by an off-the-shelf crane that can pick multiple cases at different locations before delivering the sequence of cases at a deposit point where the cases are inserted in the transport stream. Such a series of actions is called a cycle. The high-level control determines the cycles. Falcon researchers made a model to see what cycles should be given to the cranes in order to get good performance. This model forecasts the performance penalty for different number of picking tasks in one cycle. In a second variant two-sided picking is considered as well, quantifying the adverse effect of the in-sequence picking requirement on the crane's performance as a function of case size distribution. This model is a quantitative

model in a script language, in which we apply stochastic variations and many repetitions to arrive at average values and their standard deviations. The model is at 10^2 level of detail. Its construction and analysis effort amounted to less than two days. The resulting crane performance is used as input in higher-level models of the SoS, and it influenced the design of the buffer layout of the picking robot.

Case size distribution

A concise model was made to be able to reflect the case size distribution in customer orders for a limited number of concrete prospect customers, whose order data was available for a given period. The model serves to generate payload input to the crane performance model as well as to the higher-level models of the SoS. It consists of a discrete probability distribution from which samples can be drawn. The model is at 10^1 level of detail, while the time spent on it was less than a day.

ACP with "lumped control"

Automated case picking is proposed for a new market, so at the start one does not have the experience of previously built systems to rely on. Falcon researchers made more detailed models for an SoS similar to Figure [Model Fixed Sequence Picking]. The more detailed models aimed to facilitate the choice between different system-of-system concepts covering the functions "pick case from storage", "transport case", and "place case on pallet". These concepts have to be compared on different aspects. One aspect which is difficult to estimate, but highly relevant is throughput. Static models that describe performance and behavior of such SoS only exist in few situations, as dynamic interactions caused by finite buffers and phenomena such as dieback (waiting caused by full buffers) hamper such stationary approximations. Dynamic simulation models facilitate exploration of such effects.

This “lumped control” model is less abstract and more realistic than the model used to explore the impact of fixed sequencing. It works at the level of cases that are moving individually through the system. The size of cases is modeled and size distribution (dependent on the specific customer) is also taken into account. On the other hand, the modelers abstracted high-level controls to a global, constant Work-In-Progress control (keep the same amount of work in the system at all times) – this fits the regular operation of DCs in general, provided replenishment does not interfere, as the modelers assumed. This assumption is a simplification that might not hold for many concept choices; nevertheless, the results were considered useful because the effect of different layout concepts and order characteristics could be numerically studied. The model had a size of about 650 lines of code, which puts it below the 10^3 level of detail. Construction and analysis together covered about twelve days of work. With the model, some actual proposals for customers were evaluated in a very early phase of tendering for different choices of buffer sizes and transport network configurations.

ACP with detailed control mechanism on planning layers and crane breakdowns

One of the major factors impacting DC throughput and cycle time is the exception handling. E.g., failure of one or even few of the systems should have a small impact on operations. High-level controls together with the topology and concept choices determine the robustness for local failures and exceptions. The perception was that the conventional approach with a strong centralized controller lacked the capability to adjust to problems. The Falcon researchers worked on a model to support a redesign including model-driven configurability based on the hardware layout of the system. In order to do this, distributed variants of the control rules are considered, including replenishment back-orders to keep the case picking system operational and responsive (by having sufficient stock). The model of the system included the control layers to validate SoS

behavior, as well as behavior under breakdown and repair of specific components in the system. We conducted a broad set of analyses in order to build up knowledge about the effects of changing values of design parameters (e.g., component performance ratios, work-in-progress level) and their interactions. Designers typically study the effects on spreads in pallet completion times as well as the utilization levels by these models in order to gain the insight in the SoS. The implicit assumption is that spread in pallet completion time affects the integral delivery time and variation of it, and that utilization levels correlate with investment and operational costs.

The model contained over 1400 lines of code and was developed in stages. The gross development time, including tests and analyses, covered about six months. The complete analyses were collected in a technical report, which was used by system engineers to understand the characteristics of the distributed control rules and their compositions. The fact that this was a new approach for the SoS-level controls, requiring building up new insights, hardly justifies the effort put in the model and its analyses. More regular reflection on challenges at different levels was required in retrospect and would have given rise to additional focal points.

ACP in a logistics simulation tool using Enterprise Dynamics (ED) with detailed layout

Normally, a DC supplier is used to build models on a modeling infrastructure that facilitates detailed layout, dynamic behavior based on the planned high-level, and animation capabilities for a realistic visualization of the resulting DC SoS. The modelers mimic the structure of the control software in the model. The functions of this model are:

- To support the tendering process in the analysis of specification and design, e.g., concept selection, dimensioning, costing.
- To communicate with customers with a realistic and trustworthy model.

The model is highly detailed to make the visualization as realistic as possible. E.g., libraries of

product cases are available with properties and appearance so that the animation can show real cases. Traces from real good flows are used to simulate inputs and order streams, again to be as close to reality as possible. The consequence of this approach is that this model requires more effort to make or change, depending on the level of reuse that can be applied. The effort to construct and to analyze such a model is a limitation for fast exploration, but its increased accuracy (few percent level) as well as level of realism make it indispensable for the start of engineering and important customer quotations. The size of the model is estimated to be six thousand lines of code, while its (total) development time was about ten months. The development of the model went hand in hand with the development of the system control design.

AIP (Automatic Item Picking) workstation models

Heling describes several models used to support design of an automatic item picking workstation [Heling 2011].

1. Models to study the order size distribution – samples from a Poisson probability distribution and different (typical customer) scenarios including a worst-case scenario (1 item per order). This is a static, level 10^0 details, model providing quantitative insights, while it is also used in a more detailed model of the workstation.
2. A local model of the out-of-sequence arrival at the workstation that characterizes arrivals as an adapted Gamma probability density function. Domain experts recognized and verified the model results. This is also a static, level 10^0 details, model, providing quantitative insights, also used in a more detailed model of the workstation.
3. A kinematic model with only few parameters that computes travel times for the robot. It is based on formulas relating distance to maximum speeds and accelerations. The model can be plugged in other models.

4. Models to facilitate analysis of different layouts and topologies, such as lane orientation and lane flow direction in the workstation. The models help to compare cost and performance, for instance for many short lanes, or few long lanes. These models do not give absolute quantitative results, but rather support qualitative reasoning. Each of the models has only a few parameters or options, while the gross time spent on them is typically in the order of a few days.
5. A quantitative model of the cycle time for the complete pick-and-place action to explore multi-level or single-level workstation concepts, e.g., using the vertical dimension. This is an example of a model that uses the above-mentioned robot travel time model. In its turn, this model is used in the more detailed model of the AIP workstation. The model contained less than 10 parameters.
6. A more detailed model of the AIP workstation that given a configuration of the workstation facilitates analysis of the control by the information systems. This simulation can be used to compare different strategies, e.g., different customer scenarios (Models 1 and 2 above) are compared to each other. Typically, the DC designer will study many dependent variables or workstation properties in order to learn how control parameters influence the overall performance. As a result, the designers can tune control strategies according to specific customer cases. The size of the model is about 750 lines of code, while the gross time spent to develop the model as well as to perform analyses was about 4 months. For exploration and making sensible trade-offs between several system-level qualities of the workstation such as throughput and cost, this model is vital. In the scope of the overall development time of the workstation the balance between effort and impact of the model is all right.

Objectives Accomplished and Not Accomplished

The DC supplier has several means, e.g., models, to explore solution alternatives in a limited amount of time. These means help to deal with complexity and especially emergence. Models for exploration decrease the number of unexpected and undesired emergent properties. The models help to increase the desired emergent properties.

Next challenge is the embedding in the organization. Modeling for exploration is not a cookbook-based activity. It requires a lot of domain knowledge, both logistics and technical, and skills in balancing the need for details and accuracy versus exploration speed. From methods point of view, we have to realize that SoSs are moving targets where all stakeholders and systems have their own dynamics. Hence, embedding is not a static challenge, but a continuous effort.

Analysis

Analytical Findings

Reflection on Models for exploration

The conventional approach to DC design is to make a simulator with detailed layout and system information. The benefit of such modeling is that the closeness to reality helps customers in trusting the model. Disadvantage is that configuring and adapting the simulator is time and effort intensive, and that the simulator structure limits the exploration of alternatives to the preprogrammed concepts. The increasing SoS nature of the logistics chain induces more variation in the interaction between systems, with more uncertainties and an increase in

unforeseen emergent behavioral properties. Falcon researchers were complementing the traditional approach with explorative models. An explorative model typically focuses on one or a few aspects (e.g., effect of fixed sequence, crane performance, or “lumped control”). This focus on limited aspects makes it possible to simplify the model. This simplification helps to reduce the effort of making and adapting the model. However, maybe even more important, it helps designers to reason about this aspect. One of the risks of an SoS approach is that designers lose the ability to reason about different aspects, or worse, that they hide behind the complexity and unpredictability argument to not do the analysis. Figure [Models On Scale] shows the models that have been discussed on the scale with number of details. This figure shows clearly that most explorative models are typically at the 10^3 level of detail or below.

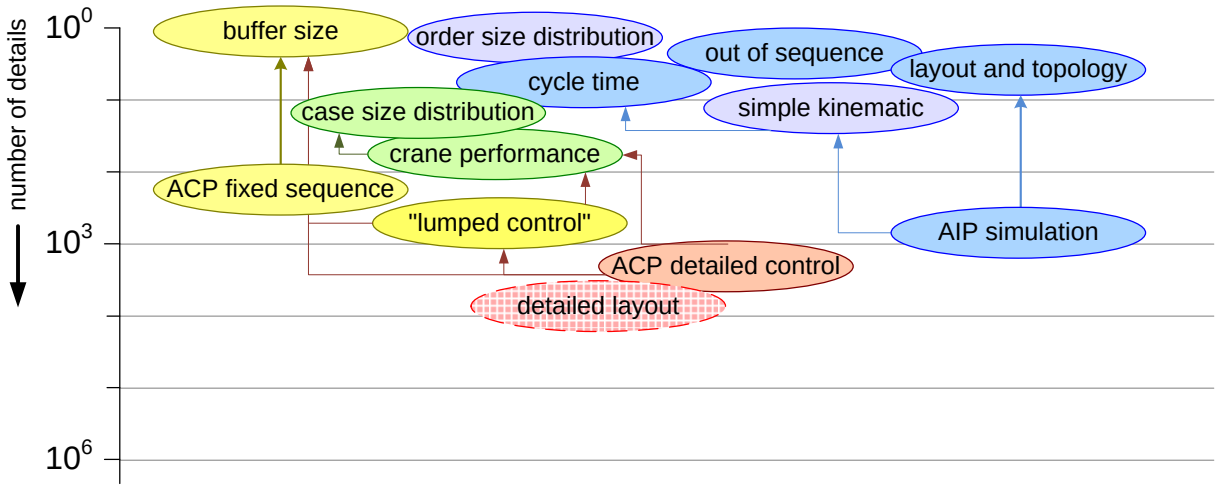

Figure [Models On Scale] The models discussed in the previous section positioned on a scale with the number of details. The dependencies between models are always qualitatively indicated.

The simplified explorative models worked well in making trade-offs and exploring concepts in the sense that the effort spent is judged to be in line with the impact of their analyses. At the

same time, designers and other stakeholders should be aware of the simplifications that went into the explorative models. These simplifications limit the validity and accuracy of the models. This nuance complicates communication with customers.

Lessons Learned

The explorative models that we have shown sometimes simplify to static models. E.g., the model relating throughput and buffer sizes is ultimately a mathematical formula that entirely ignores the dynamics of the goods flow. Some of the models are dynamic and do take variations in goods flow or systems into account. Dynamic models are inherently more detailed. The inclusion of the dynamics allows analysis of effects such as peak loads, resonances, and disturbances. Dynamic models tend to be closer to reality. These properties being advantageous, they also invite or provoke to put more details into them than required for a specific analysis. This is a trap of using dynamic models, encountered by ourselves as well in a single case. The remedy to this is a regular reflection on all levels of the SoS, i.e., explicitly zooming in and out. Altogether, it is very well possible to apply dynamic models in explorative phases, provided the goals of the models are sufficiently crisp, having a limited scope.

Similarly, we have shown qualitative models; models that give a relative answer (e.g., better, faster, more). Quantitative models give absolute answers (e.g., cases per hour), but when the systematic errors are known to be large, they still can be used to produce relative answers. Qualitative models facilitate reasoning, even when the system properties are still unknown. Qualitative models can help to understand concepts and trade-offs, before lots of effort is spent in quantitative understanding. In later phases quantitative models are required to facilitate concept and technology choices.

Best Practices

Multi-level modeling

The models that we have discussed are mostly multi-disciplinary models. Allocation of functionality to specific disciplines, such as mechanical, electrical, or software is not yet relevant. Most models belong to a particular system level (e.g., DC, pick and place, or crane or robot). A model at one level can "feed" models at a next level. For example, the crane performance model can provide input to pick and place models. Such model connections are often not literal; the results at one level can often be simplified or abstracted at a next level manually. The virtual world of exploratory models can be viewed as a micro-cosmos of SoSs.

Multi-view modeling

Most models need multiple views to be explained and created. We especially see the use of physical, functional, and quantified views as generic views. Most stakeholders associate with physical models. Physical models are concrete and tangible. Functional models show how systems work. Functional models often show dynamics by some flow (e.g., of goods, energy, or information). The dynamics of functional models complements the partitioning oriented view of physical models. Quantification facilitates analysis and reasoning, and makes discussions more concrete. These three generic views are applicable throughout all levels of systems from enterprise down to component. These generic views often need additional complementary views for further communication and discussion.

Abstraction and Simplification

A higher level of abstraction means to represent essential properties in a fundamental way without losing their characteristic dependencies, and to leave out non-essential properties altogether. A challenge for systems engineers is to find proper levels and ways of abstraction.

They need to relate models at various abstraction levels to reason about specification, design, and engineering challenges. We hope that discussion of several examples will illustrate the potential utility of modeling for SoS.

Replication Prospects

The approach using models to explore is one of the main methods in architecting SoSs. The T-AREA-SoS project identified multi-level modeling and conceptual modeling (*insight through simple models* [T-AREA-SoS 2013]) as key research topics. Architects have used models for exploration at system and enterprise level in the past. The transition to SoS level amplifies the need for applying the best practices: multi-level modeling, multi-view modeling, and abstraction and simplification.

Summary and Conclusion

DCs are typical SoSs, embedded in a logistics enterprise. Stakeholder key performance parameters can be analyzed at all layers, and conversely, technology and concept properties can be studied at all layers. Designers need sufficient insight in the relation between key performance parameters and technology and concept properties to come to a robust, affordable, and well-performing DC. We assert that explorative models, e.g., models that can be made with limited effort and time, are necessary to cope with the complexity, dynamics, and uncertainties of the SoS world. These models focus at different levels of the system hierarchy, have limited and clear goals, and regularly use each other's results. Frequent and explicit reflection on the modeling activity is one way to avoid the trap of putting more details into the models than they require for achieving their goals.

We need to see more of their use to accelerate the evolution of the logistics enterprise, which can

be enabled by deepened insights across several system levels, starting from traditional logistic components such as conveyors, which are well known to most DC suppliers, all the way up to expected trends in the world of global goods distribution.

Suggested Future Work

Further Questions for Discussion

Imagine a distribution center for a supermarket chain serving the city of New York. For this distribution center make the following estimates:

- The number of different products that flow through such center
- The order of magnitude of the amount of goods flowing through the center (in quantity, volume, weight, or value)
- What goods require special treatment, e.g., cooling, handling fragile goods, etc.
- The number of suppliers of goods and the number of super markets served
- Show for one specific product what happens to it from supplier to consumer

Additional Research

We have observed that experienced systems architects and engineers make explorative models by nature. However, many “fresh” systems engineers tend to dive and disappear into detailed models; they struggle with fundamentals and get overwhelmed by complexity. What guidelines and methods can we provide to assist less experienced systems engineers to make explorative models?

References

[Ashby 2001] **W. Ross Ashby, "**The Law of Requisite Variety**,"** in the Principia Cybernetica Web**, 2001,** http://en.wikipedia.org/wiki/William_Ross_Ashby

[Falcon 2011] **Falcon project**, 'System-of-systems' performance and reliability in logistics, http://www.esi.nl/falcon/, accessed October **2012**

[Fisher 2006] **David A. Fisher,** "An Emergent Perspective on Interoperation in Systems of Systems", technical report CMU/SEI-2006-TR-003, 2006

[Hamberg 2012] **Roelof Hamberg, Jacques Verriet** (eds.), Automation in Warehouse Development, Springer, London, UK, **2012**

[Heling 2011] **J.W.E. Heling**, Design of an Automated Item Picking Workstation, Master Thesis at Eindhoven University of Technology, June **2011**

[Kopetz 2011] **Hermann Kopetz,** "Real-Time Systems: Design Principles for Distributed Embedded Applications", Springer, 2011

[Muller 2011] **Gerrit Muller**, Systems Architecting: A Business Perspective, CRC Press, Boca Raton, Fl., USA, **2011**

[Roode 2011] **V. Roode**, Exception handling in automated case picking. SAI technical report, Eindhoven University of Technology, Eindhoven, **2011**

[T-AREA-SoS 2013] T-AREA-SoS project, Agenda for expert workshop, January 2013, https://www.tareasos.eu/ accessed February **2013**