Modeling and Software

Introduction to modeling

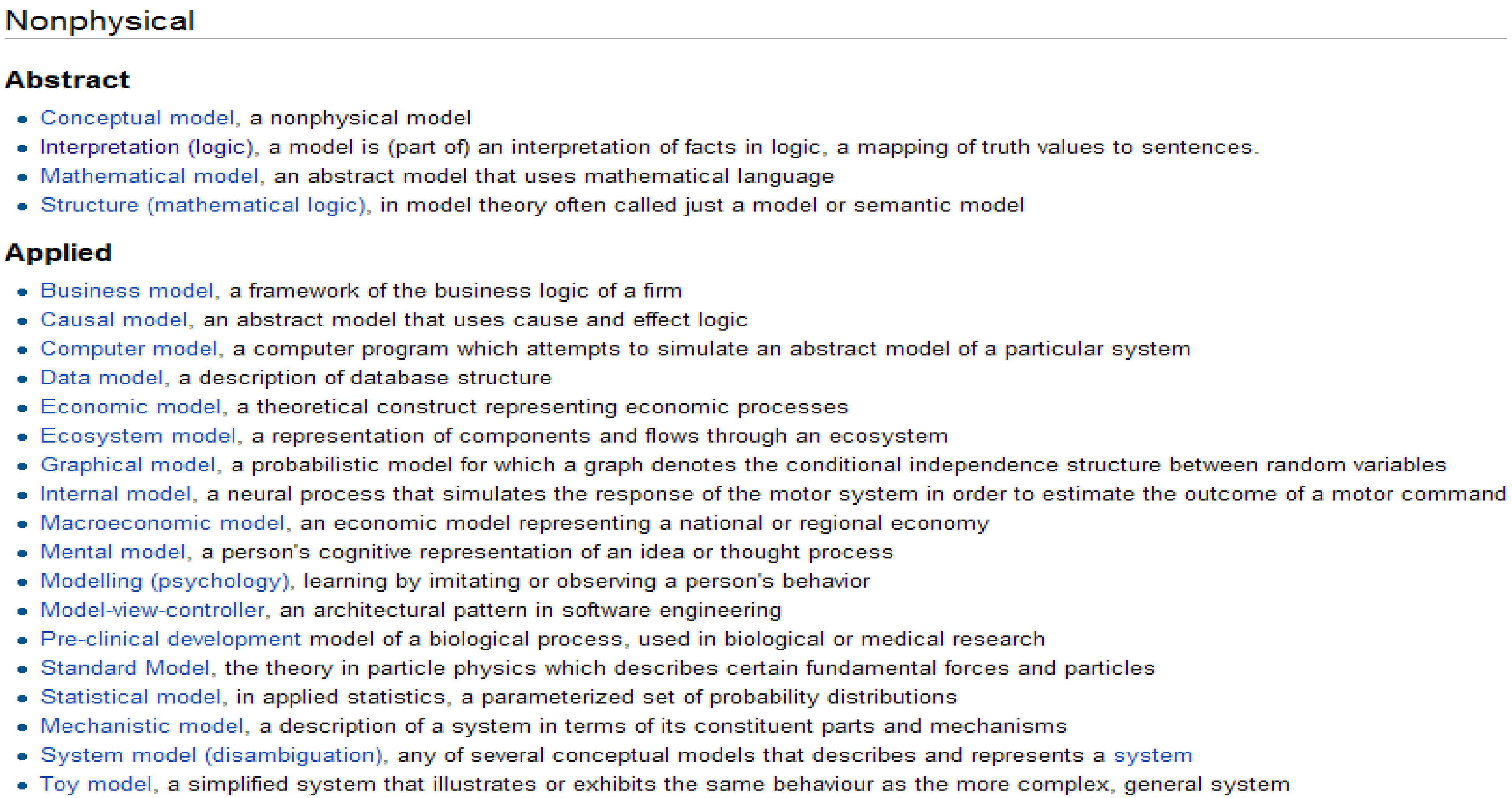
Visual modeling

Software complexity

Software architecture

Model

- ***Model***: **textual or graphical description of a system or process (existing or such to be crated);** in http://en.wikipedia.org/wiki/Model



Nonphysical

Abstract

- Conceptual model, a nonphysical model
- Interpretation (logic), a model is (part of) an interpretation of facts in logic, a mapping of truth values to sentences.
- Mathematical model, an abstract model that uses mathematical language
- Structure (mathematical logic), in model theory often called just a model or semantic model

Applied

- Business model, a framework of the business logic of a firm
- Causal model, an abstract model that uses cause and effect logic
- Computer model, a computer program which attempts to simulate an abstract model of a particular system
- Data model, a description of database structure
- Economic model, a theoretical construct representing economic processes
- Ecosystem model, a representation of components and flows through an ecosystem
- Graphical model, a probabilistic model for which a graph denotes the conditional independence structure between random variables
- Internal model, a neural process that simulates the response of the motor system in order to estimate the outcome of a motor command
- Macroeconomic model, an economic model representing a national or regional economy
- Mental model, a person's cognitive representation of an idea or thought process
- Modelling (psychology), learning by imitating or observing a person's behavior
- Model-view-controller, an architectural pattern in software engineering
- Pre-clinical development model of a biological process, used in biological or medical research
- Standard Model, the theory in particle physics which describes certain fundamental forces and particles
- Statistical model, in applied statistics, a parameterized set of probability distributions
- Mechanistic model, a description of a system in terms of its constituent parts and mechanisms
- System model (disambiguation), any of several conceptual models that describes and represents a system
- Toy model, a simplified system that illustrates or exhibits the same behaviour as the more complex, general system

The Role of a Model

- Models in human life and engineering activities
- History of modeling – drawings describing static character of an object/a system, and its behavior (dynamics character)
- Role of modeling at:
  - Planning of activities
  - Project evaluation – as time and money
  - Work load distribution and allocation of resources

Early humans brainstorming on a proto-whiteboard (by Tom Gullion)

Model and Modeling

- ***Model*: (mathematical) presentation of structure and processes of a given system (used for analysis and planning)**

- ***Modeling*: process of describing of the system by means of its model (mathematical or based on imitation) and simulation of system activities by means of applying the model on a data set**

Modeling

- Phases in model development
- Iterations in model development
- Requirements to the end product – about functionality, external design, performance and reliability
- Models are not final – usually, they change during the project time in order to reflect new approaches and experiences

Types of Models

- **Static models – describe system structure, e.g. E/R data model. Data types – analogues, discrete, or hybrid data.**
- **Dynamic models – describe system behavior. They depending on the chosen apparatus and can be:**
  - **Analytical – describe by differential equations**
  - **Imitation models – usually, they are built by a visual language such as Petri nets, OMT, OBLOG, UML, etc. We cannot describe all the system details visually. Thus, we have to look for a *balance between visual and textual description*.**

Appliance of Models

Practical models should be:

- **describing the system in a correct way**
- **consistent – different views should not describe things being in conflict each other**
- **easy to be explained to and understood by other people – *as simple as possible but not simplified***
- **easy for updates and maintenance**
- **in a form suitable for transfer to other people**

A Visual Model

***Visual modeling*:**

- **a way of thinking about problems using practical and graphical models derived by ideas of the real world**
- **graphical modeling presenting treated system in two- and three dimensional way.**

***Visual modeling language* – manipulates visual information, i.e. presents systems through graphical (icon-based) and textual expressions according given dimensional grammar[1]. Visual languages for specification and for programming.**

[1] Microsoft Visual Basic and Visual C++ are not visual languages

Visual Models

Graphical abstractions describe the essence of a complex problem or of a structure by filtering immaterial details. In such a way, they do the problem easier for understanding and presentation. We can compare them with the architectural approach – drawn models.

When building a complex system, developers should do:

- Extract different views about the system
- Build models using precise symbols (notations)
- Verify/validate that the models satisfy system requirements
- Add details in order to transform models into implementation, step by step

Modeling Method

A method is a generic guideline for realization of modeling; contains specific knowledge for various cases such as patterns and conventions.

A ***method*** is a mixed bag of guidelines and rules, including the following components (*Dr. James Roumbaugh, 1995*):

- ***modeling concepts***
- ***views and notations***
- ***development process***
- **hints and rules-of-thumb**

More precisely, a **method** has to include:

- A set of fundamental ***modeling concepts*** for capturing semantic knowledge about a problem and its solution. The modeling concepts are independent of how they are visualized. They are the inputs for semantic tools, such as code generators, semantic checkers, and traceability tools.

- A set of ***views and notations*** for presenting the underlying modeling information to human beings which allow them to examine and modify it. Normally the views are graphic, but multimedia interfaces are feasible with current technology. Graphic views use geometric arrangement and graphic markers to highlight portions of the semantic information. Usually each view shows only a part of the entire semantic model.

- A step-by-step iterative ***development process*** for constructing models and implementations of them. The process may be described at various levels of detail, from the overall project management down to the specific steps to build low-level models. The process describes which models to construct and how to construct them. It may also specify measures of goodness for evaluating proposed designs.
- A collection of **hints and rules-of-thumb** for performing development. These are not organized into steps. They may be applied wherever they make sense. The concept of ***patterns*** is an attempt to describe case-based experience in a uniform way. Patterns represent specific design solutions to recurring problems. These may apply at various levels of detail, from large-scale architecture down to low-level data structures and algorithms.

Why do we model?

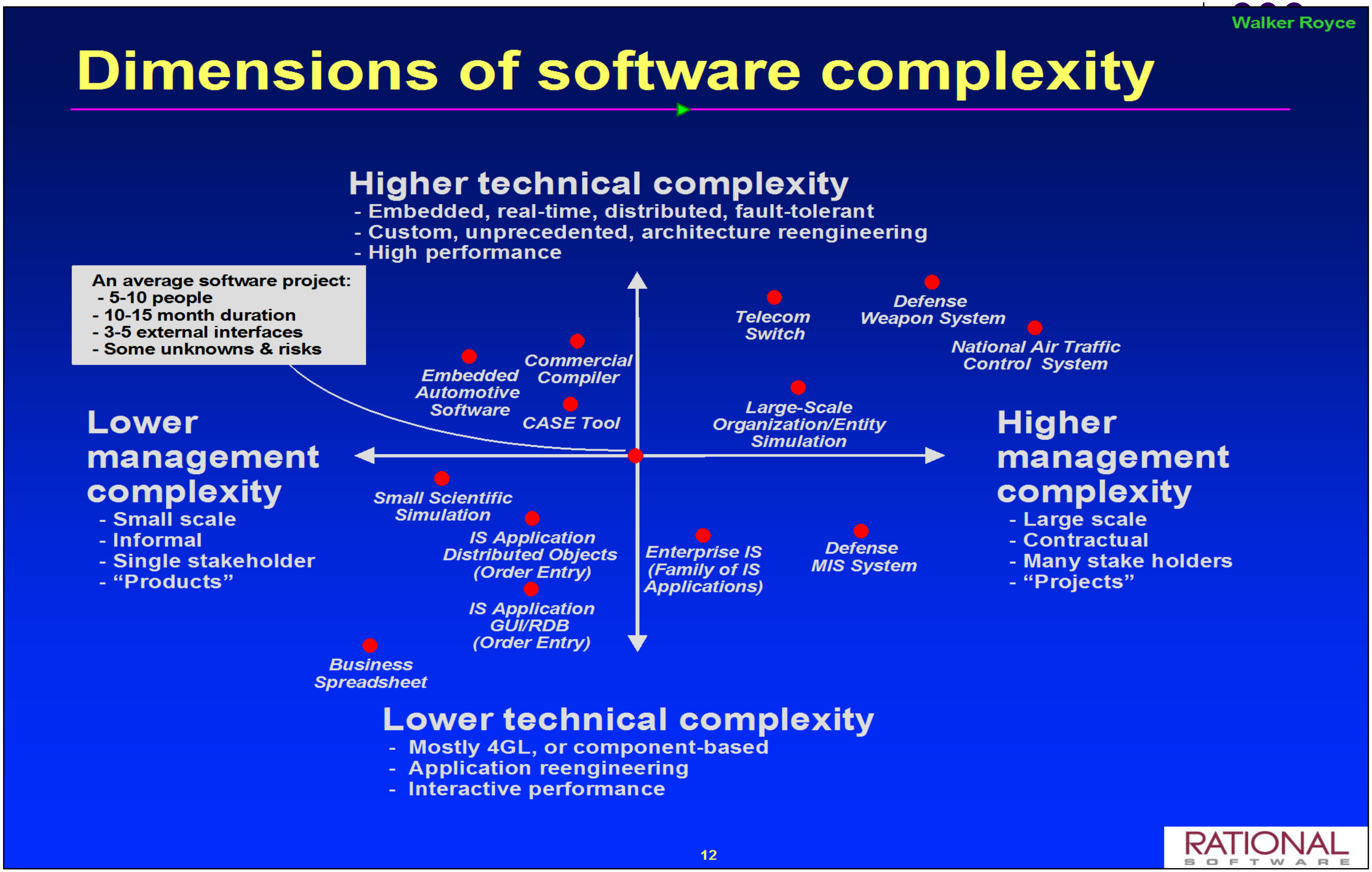
- **Models help in organizing, visualizing, understanding and creation of complex information systems.**
- **Two challenges for software developers:**
  - **Business environment is highly concurrent and dynamic**
  - **Increasing system complexity**

Models help meeting them.

Conceptual diagrams help software development

By modeling, we can represent visually using industry-standard diagrams many issues of software development:

- Overall architecture of the system
- Business requirements
- System boundaries and dependencies
- System complexity
- Flow of information through a system
- Data model organization and structure



Walker Royce

Dimensions of software complexity

Higher technical complexity
- Embedded, real-time, distributed, fault-tolerant
- Custom, unprecedented, architecture reengineering
- High performance
An average software project:
- 5-10 people
- 10-15 month duration
- 3-5 external interfaces
- Some unknowns & risks
Telecom Switch
Defense Weapon System
National Air Traffic Control System
Commercial Compiler
Embedded Automotive Software
CASE Tool
Large-Scale Organization/Entity Simulation
Lower management complexity
- Small scale
- Informal
- Single stakeholder
- "Products"
Higher management complexity
- Large scale
- Contractual
- Many stake holders
- "Projects"
Small Scientific Simulation
IS Application Distributed Objects (Order Entry)
Enterprise IS (Family of IS Applications)
Defense MIS System
IS Application GUI/RDB (Order Entry)
Business Spreadsheet
Lower technical complexity
- Mostly 4GL, or component-based
- Application reengineering
- Interactive performance
12
RATIONAL SOFTWARE

Software Architecture

Complexity determines software architecture – a set of important decisions about software system organization.

- **Choice of structural elements and interfaces for system composition**
- **Behavior – determined by elements' collaboration**
- **Composition of the chosen structural elements and their behavior into a larger system**
- **Architectural style**

Software Architecture - More

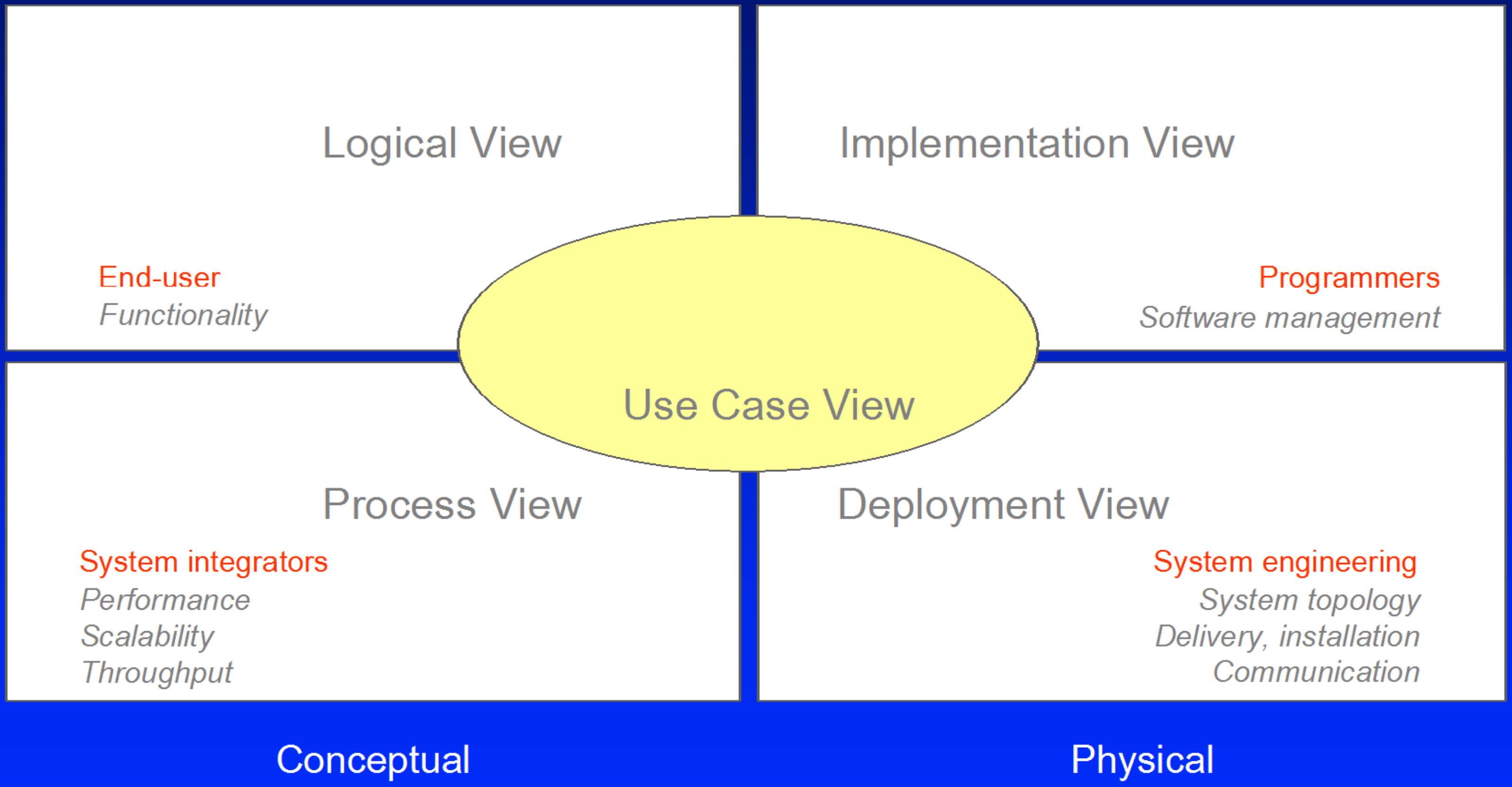
Moreover, the software architecture includes:

1. **Usability**
2. **Functionality**
3. **Performance**
4. **Flexibility**
5. **Reuse**
6. **Economical and technical restrictions and their balance – “*good, fast, cheap – choose only two!*”**

Architectural Style

- ***Defines a family of systems by means of pattern for structural organization. In other words, it defines:***
- **Component dictionary and types of connecting elements**
- **Set of restrictions and how we can combine them**
- **One or more semantic models specifying how to determine common system properties based on the properties of its building blocks.**

Representing System Architecture



26

RATIONAL SOFTWARE