User Interactions and their impact on DSL Design: An interactive case study for Radiation Oncology Workflow

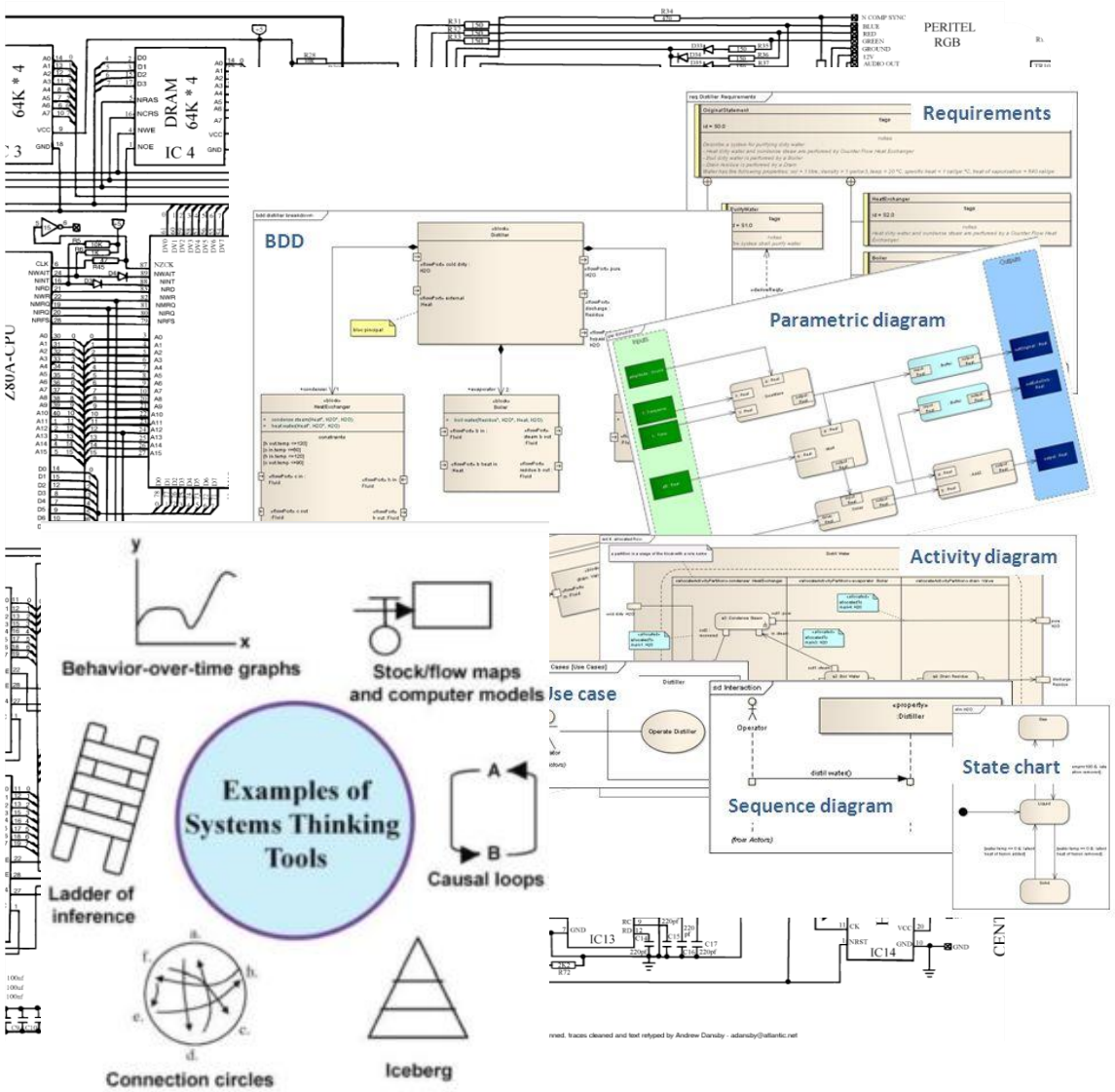
John Wolfgang / Frank van der Kruijssen

STAF 2019 – Industry Day

ICT GROUP

ICT smarter solutions

ICT GROUP Netherlands, Belgium, Germany, Bulgaria, Sweden



Frank van der Kruijssen MSc

System Architect

John Wolfgang PhD

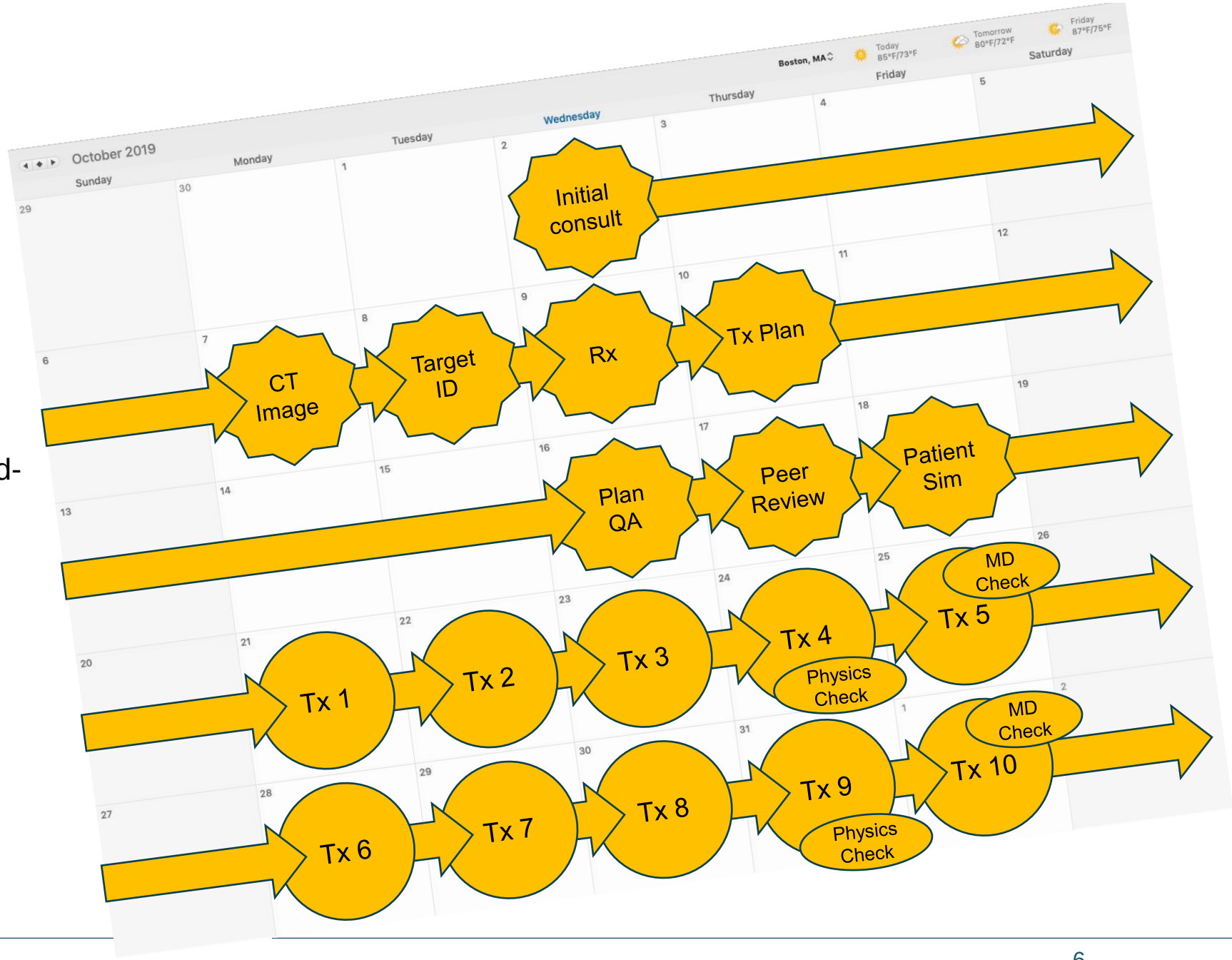
Radiation Oncology Physicist

MGH and Partners Healthcare

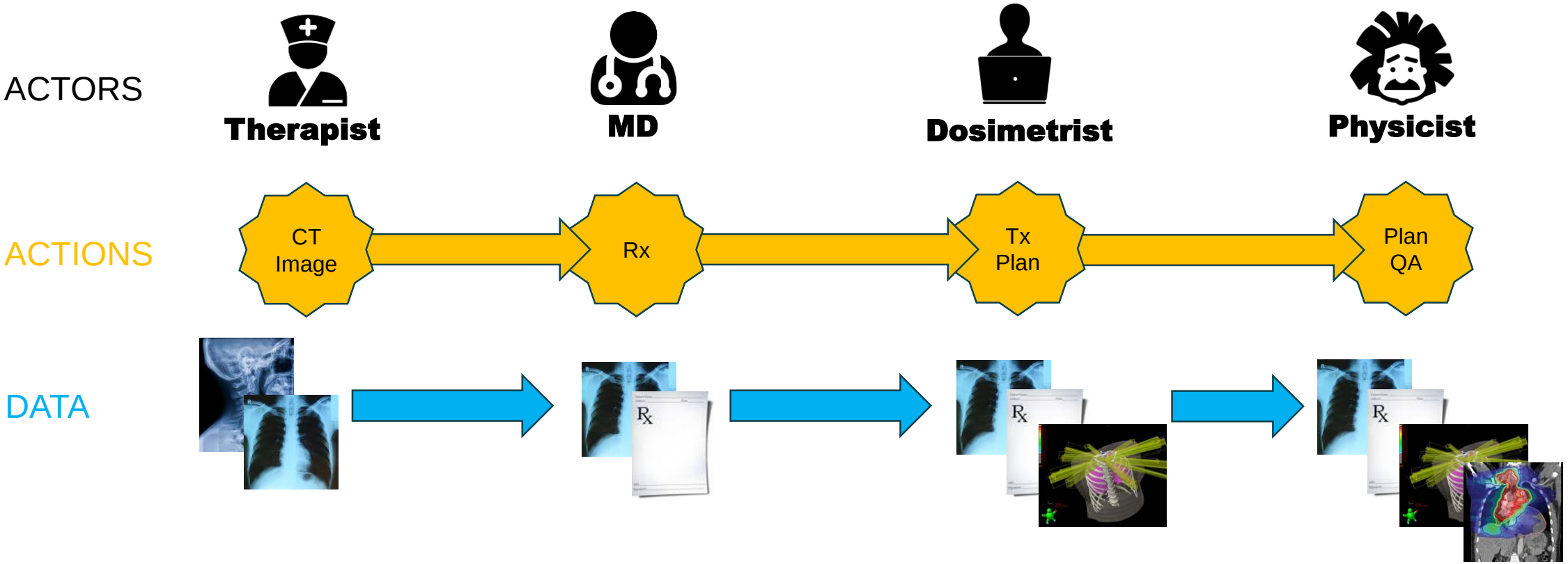
- Massachusetts General Hospital is part of the Partners Healthcare network
- Oncology
  - Surgical
  - Medical (Chemotherapy)
  - Radiotherapy
- MGH Radiation Oncology
  - 2500 patients/year
  - Approx 64,000 treatments/year
  - Photon
  - Proton
  - Brachytherapy
  - Intra-operative radiotherapy
  - Many different techniques for each

Patient's Path Through Radiation Oncology

- Multiple activities, each dependent on prior events
- Extended timeframe
  - 1 week – 2 months
- Multiple actors / activity hand-off
  - Radiation Oncologist
  - CT Therapist
  - Radiation Therapist
  - Dosimetrist
  - Physicist

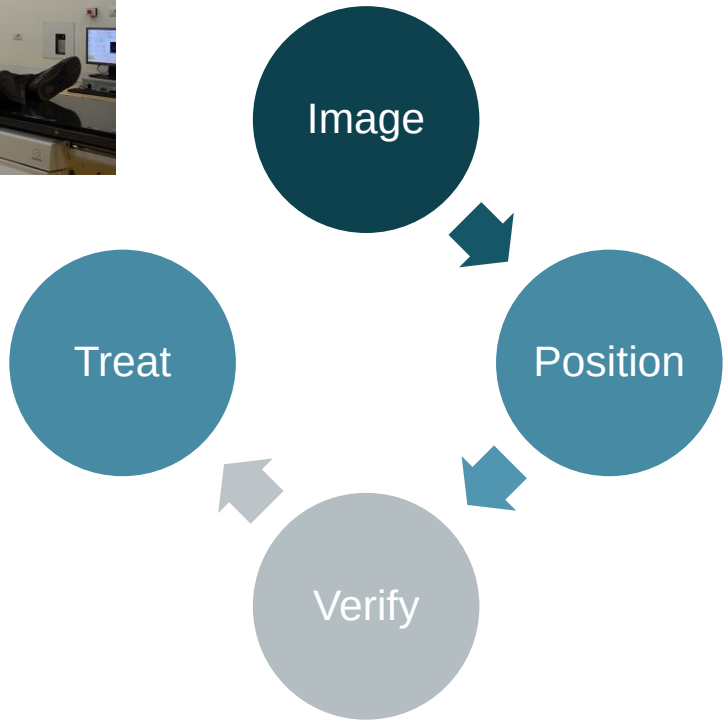


Data and Workflow

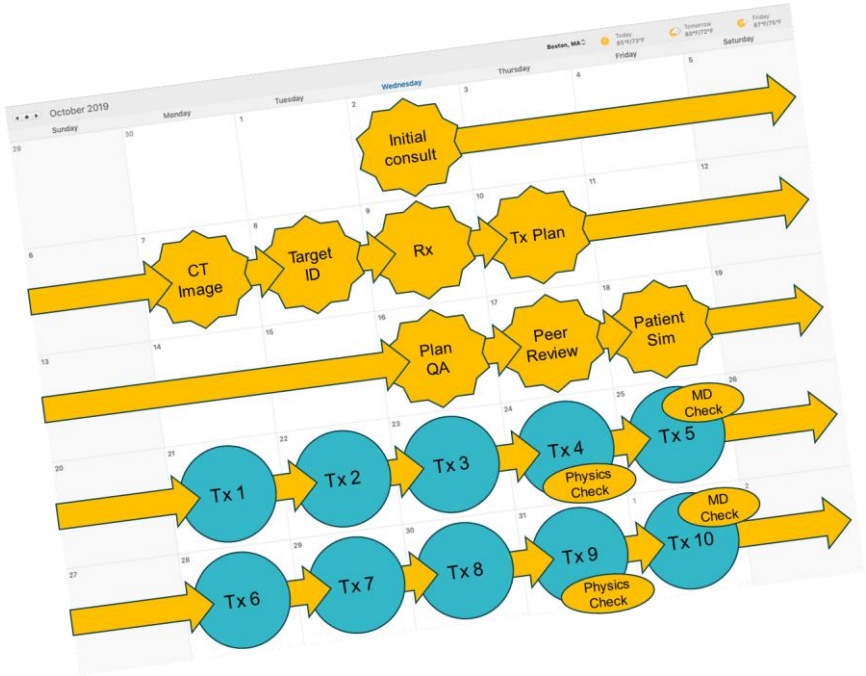


- Actors perform actions that generate data
- Data must be used by next step in workflow
- Require correct data for each operation
  - Versioning, variations
  - e.g. Many CT images taken, require one for planning

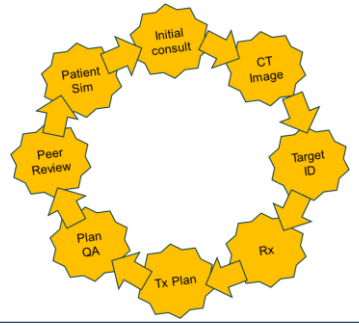
Daily Treatment Workflow

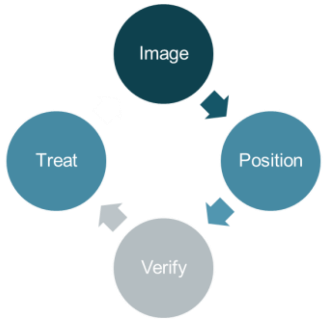


- Treatment Workflow
  - Image patient position
  - Adjust patient position
  - Verify patient position (re-image)
  - Treat patient (delivery radiotherapy plan)



A Workflow and Data Management system

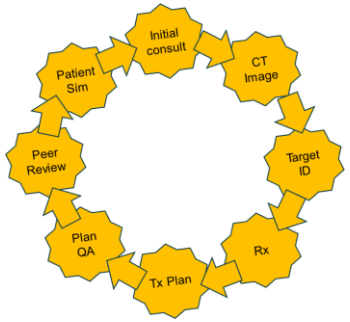


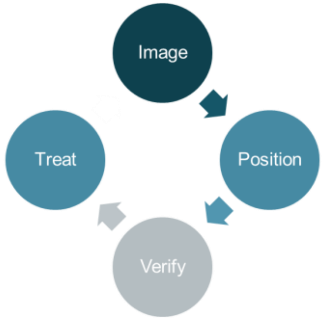


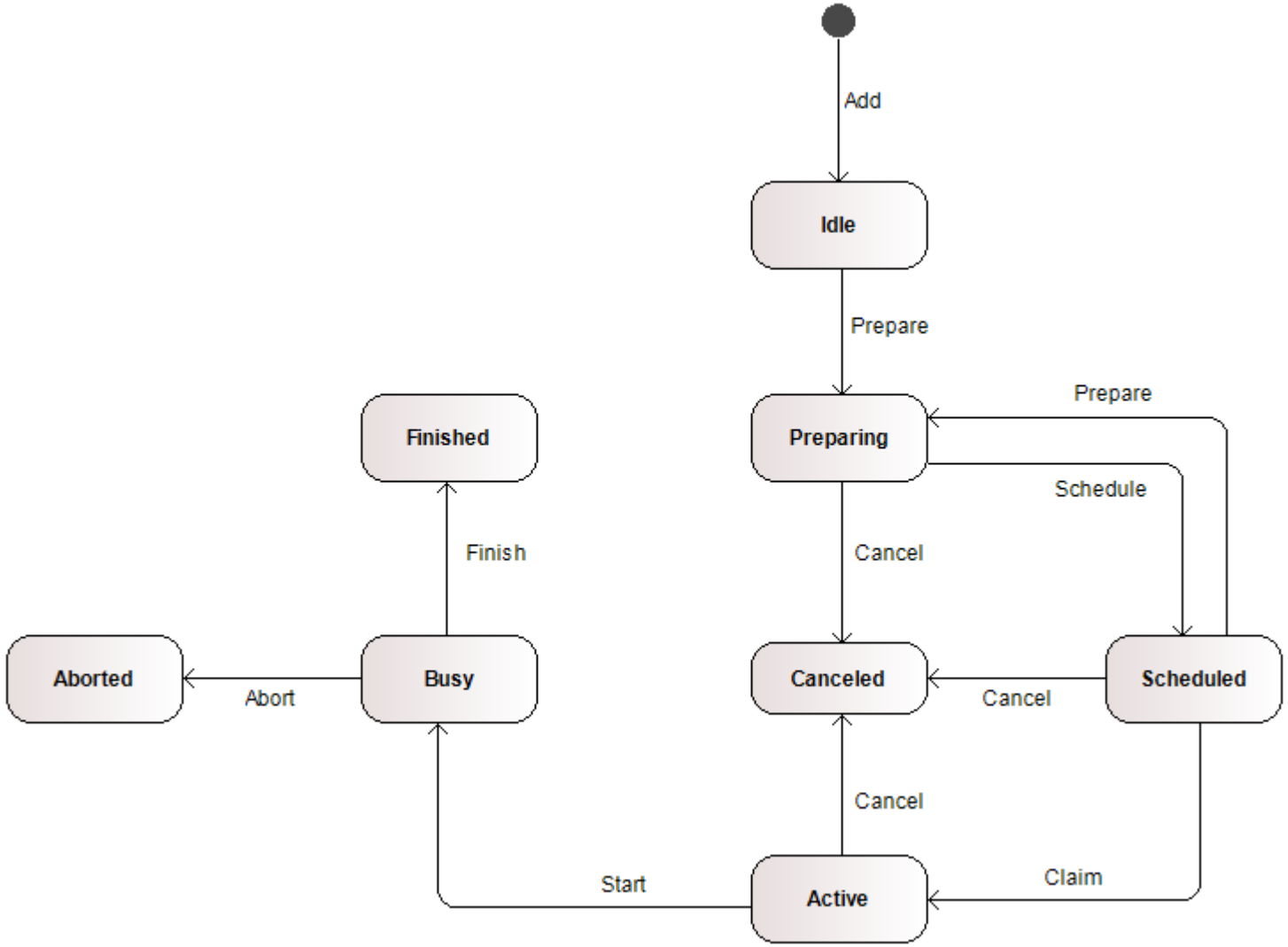
Session FSM

Simplified

- Manages the workflow
  - The scheduling and
  - preparation part
  - The execution part
  - and exception handling
- 8 states in total



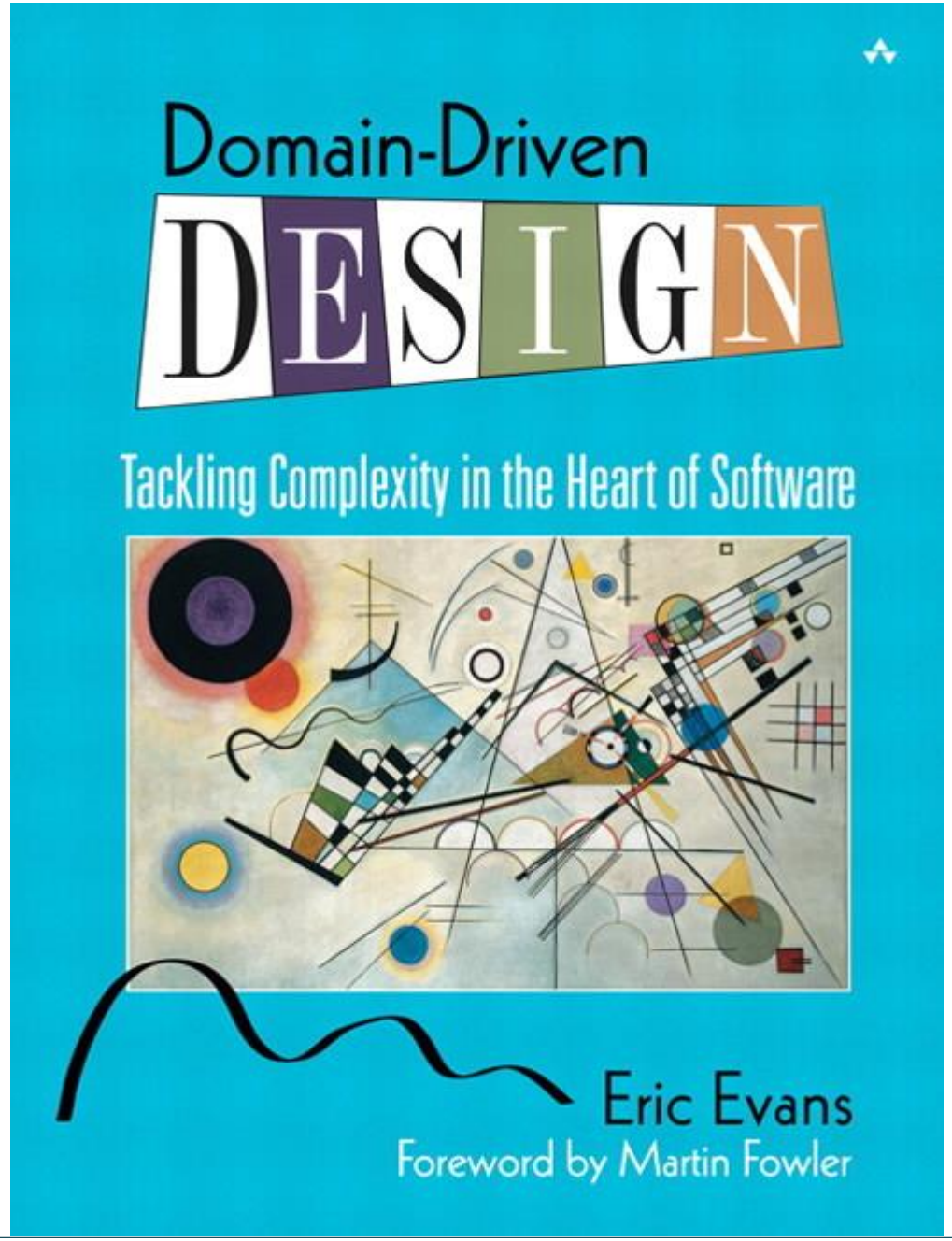




Domain-Driven Design

- Introduced in 2004 by Eric Evans
- Helps in structuring the discussion to find and define terminology
- which serves as the basis of a shared, ubiquitous language

- System engineer and domain expert both have the same understanding of the language
- which serves as the basis for the system design
- thereby tackling the complexity in the heart of the software



Domains of our system

- **Patient**

  Patient identification and demographics
- **Patient record**

  Departmental treatment record
- **Treatment Definition**

  Specific information to manage a treatment
- **Treatment Management**

  Workflow and Data flow
- **ERP**

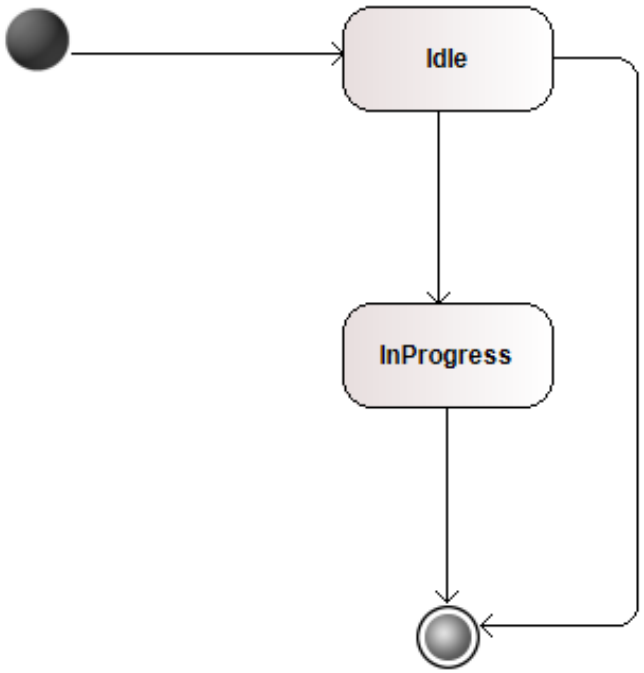
  Resource management and planning
- **QA**

  Audit trailing

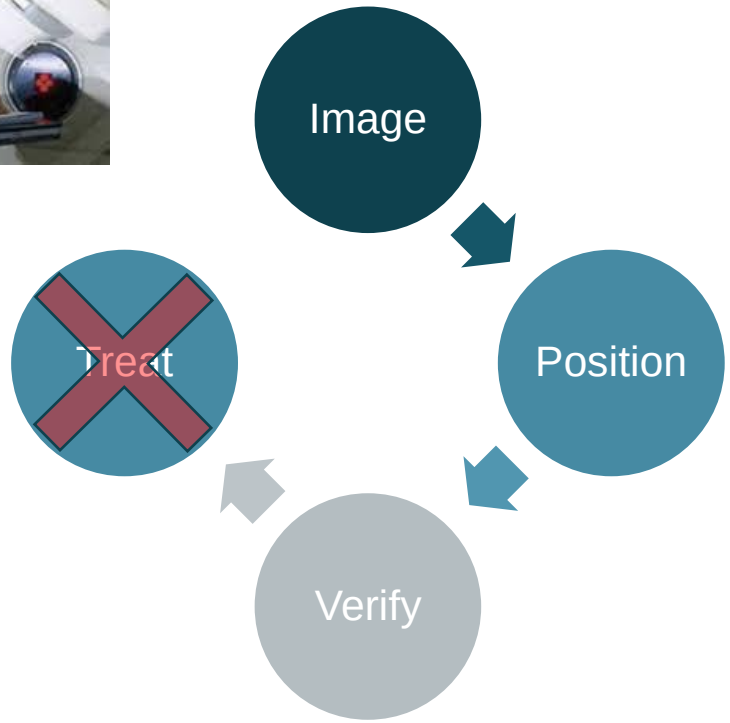
Treatment Management

Treatment Management

- A **Session** now has only two simple states:
  - *Idle*
  - *InProgress*



Workflow Behaviors



- Treatment Workflow
  - Image patient position
  - Adjust patient position
  - Verify patient position (re-image)
  - Treat patient (delivery radiotherapy plan)
- But things do not always go according to plan …
  - Patient gets sick during treatment
  - Delivery system fails during treatment
- How to resume?
  - Data continuity issues
  - Activity continuity issues

    Within session

    Next session (tomorrow)

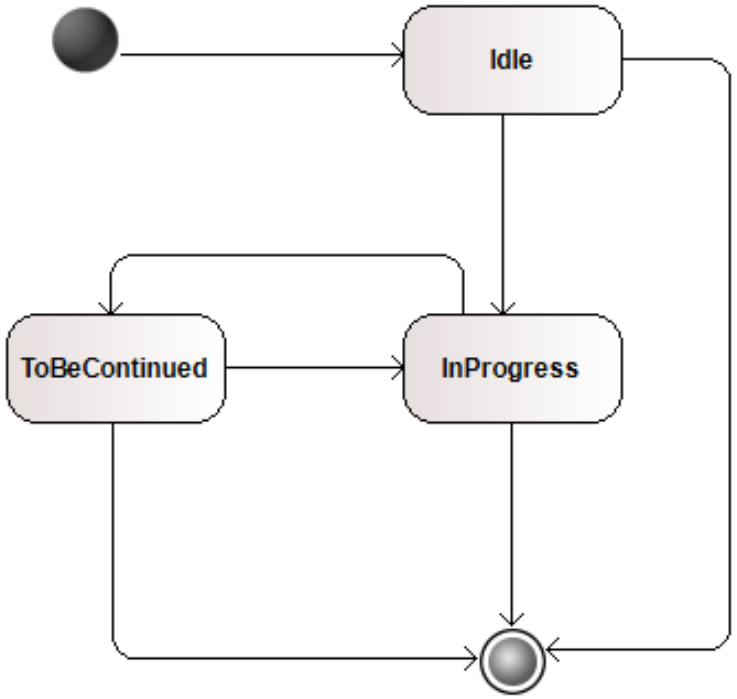
Treatment Management

Treatment Management

- A **Session** has two simple states:
  - *Idle*
  - *InProgress*

- Continuation will introduce a new state:
  - *ToBeContinued*

- We can go there from *InProgress*
- We can go back to *InProgress*
- Or we can immediately go to the end state



Simple… or maybe not?

Keeping domains clean

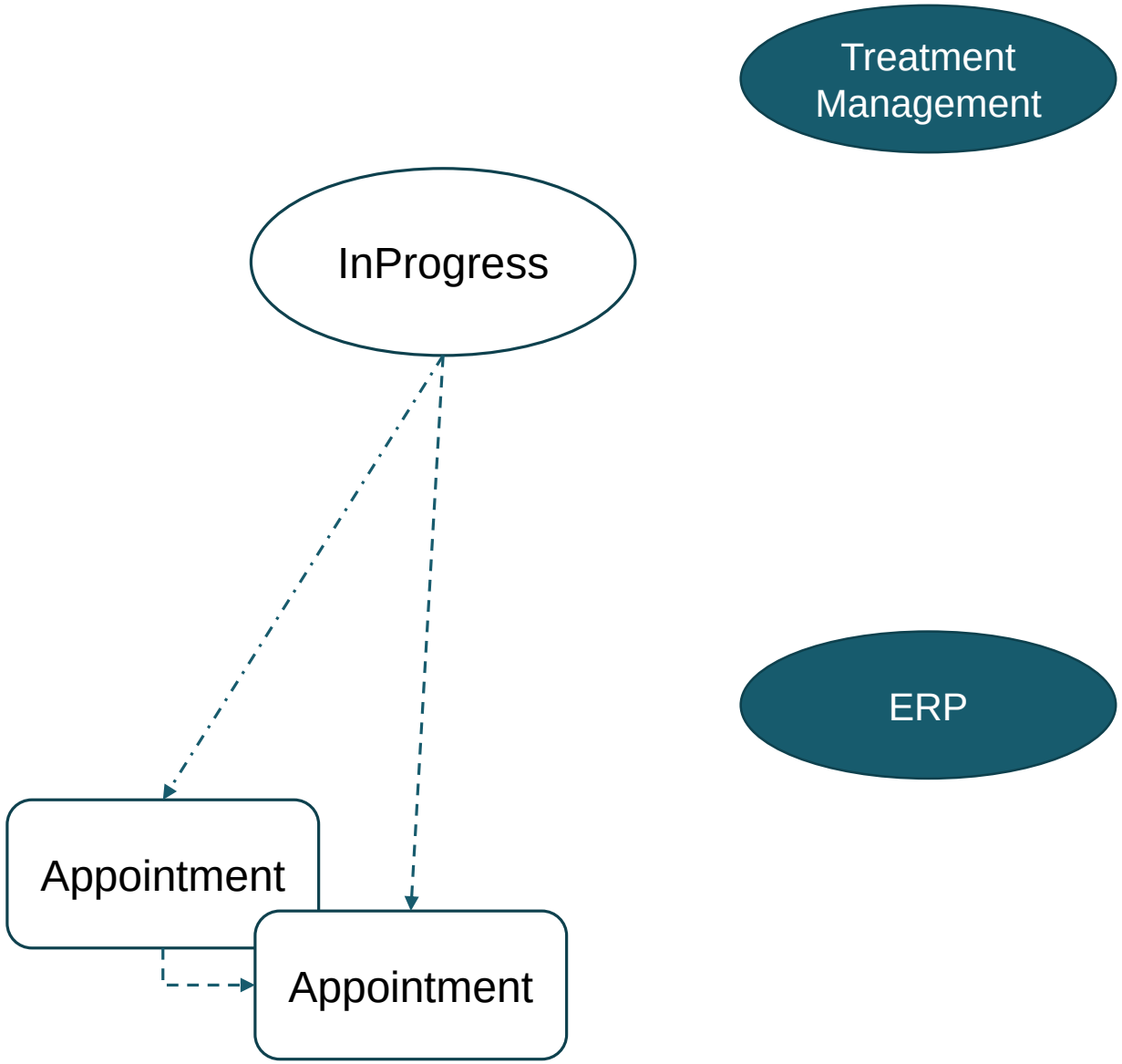
Introducing a new state increases complexity

- Larger state space
- All clients of this state machine will need to deal with the extra state

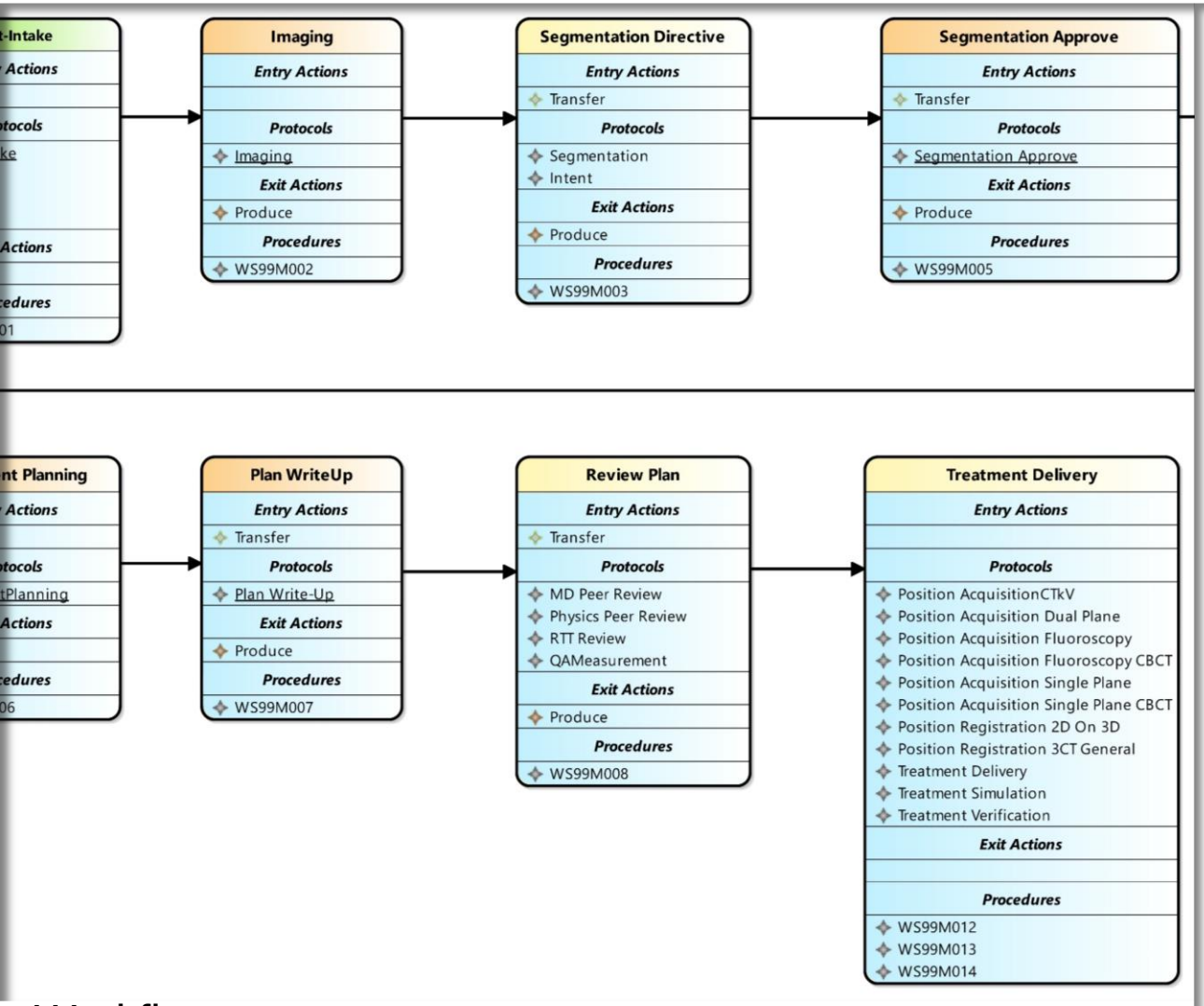
Instead,

- Adding a new domain, or
- combining the domains differently

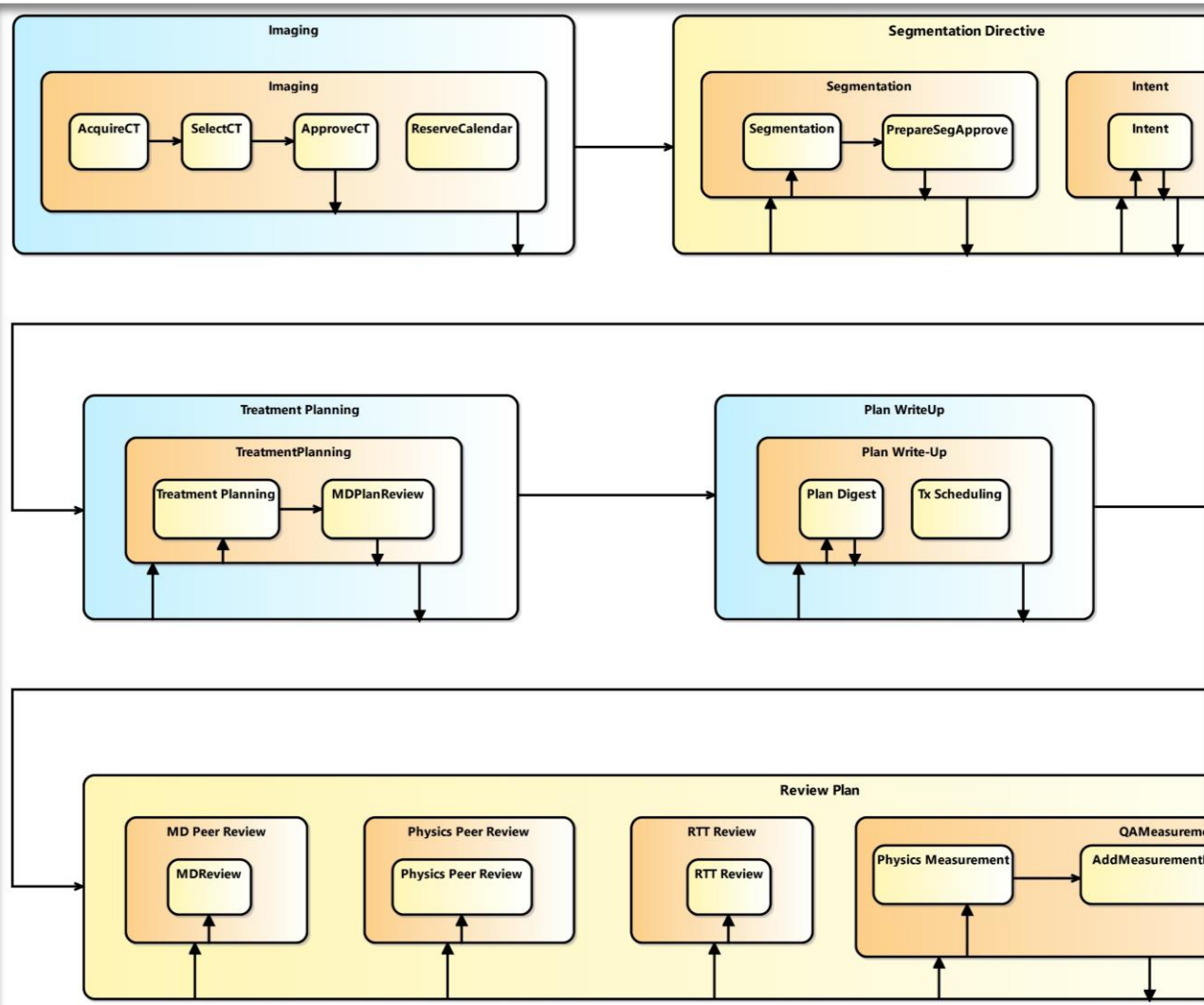
will not (or hardly) increase complexity



Workflow as DSL



Workflow

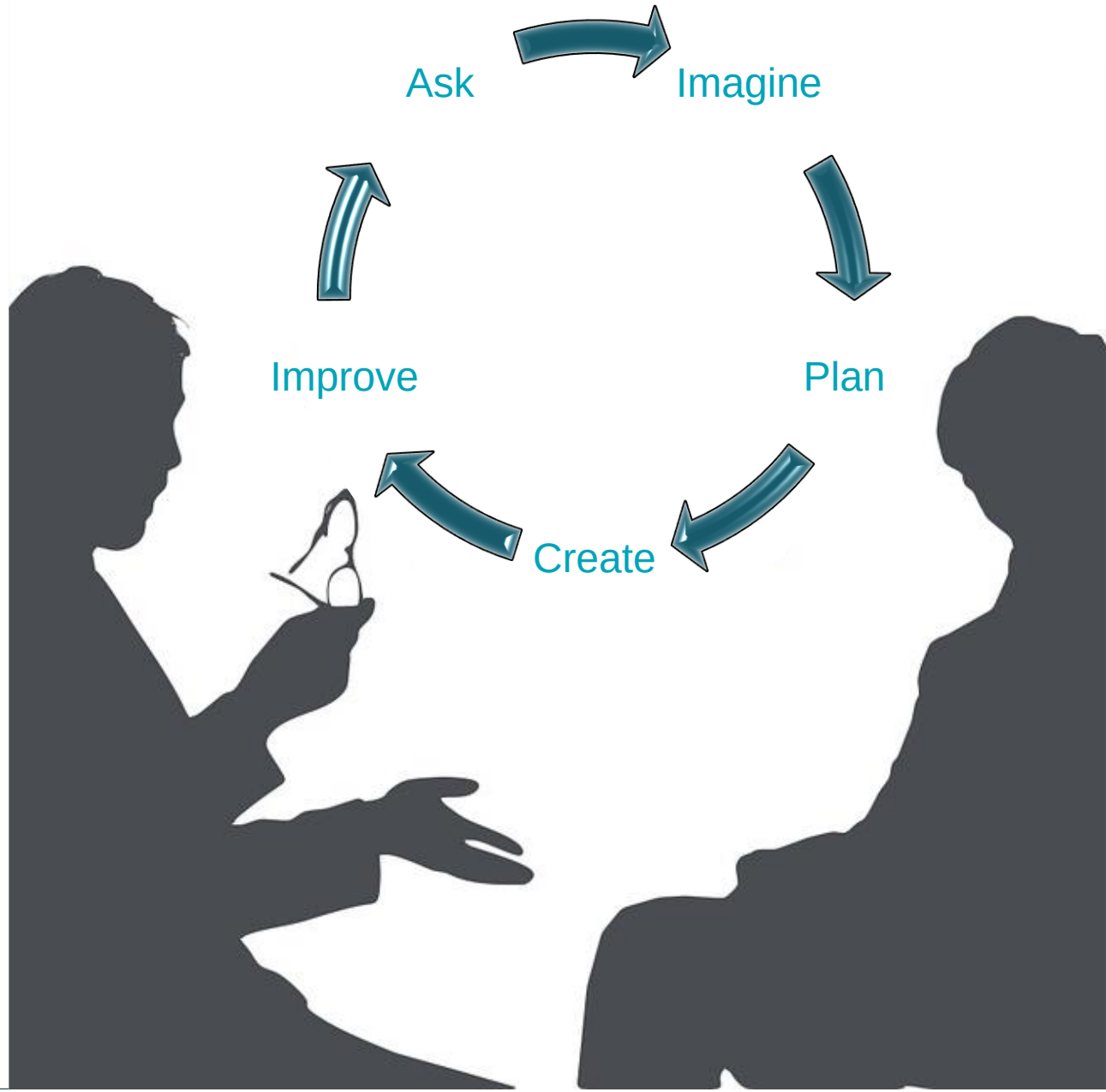


Dataflow

Language through dialogue

The Engineering Design Process *

- Ask questions about the problem, approach and constraints
- Imagine the situation and brainstorm
- Plan the language. Start identifying elements
- Create the language
- Improve by considering alternative problem definitions, go back and ask questions



* Engineering is Elementary, Museum of Science, Boston, MA, USA

Shared language

- Domain expert and system engineer jointly develop a shared language
  - No translations
  - No loss of information
- This language is used to define the system and the algorithms
  - New features are well understood by both the domain expert and the system engineer

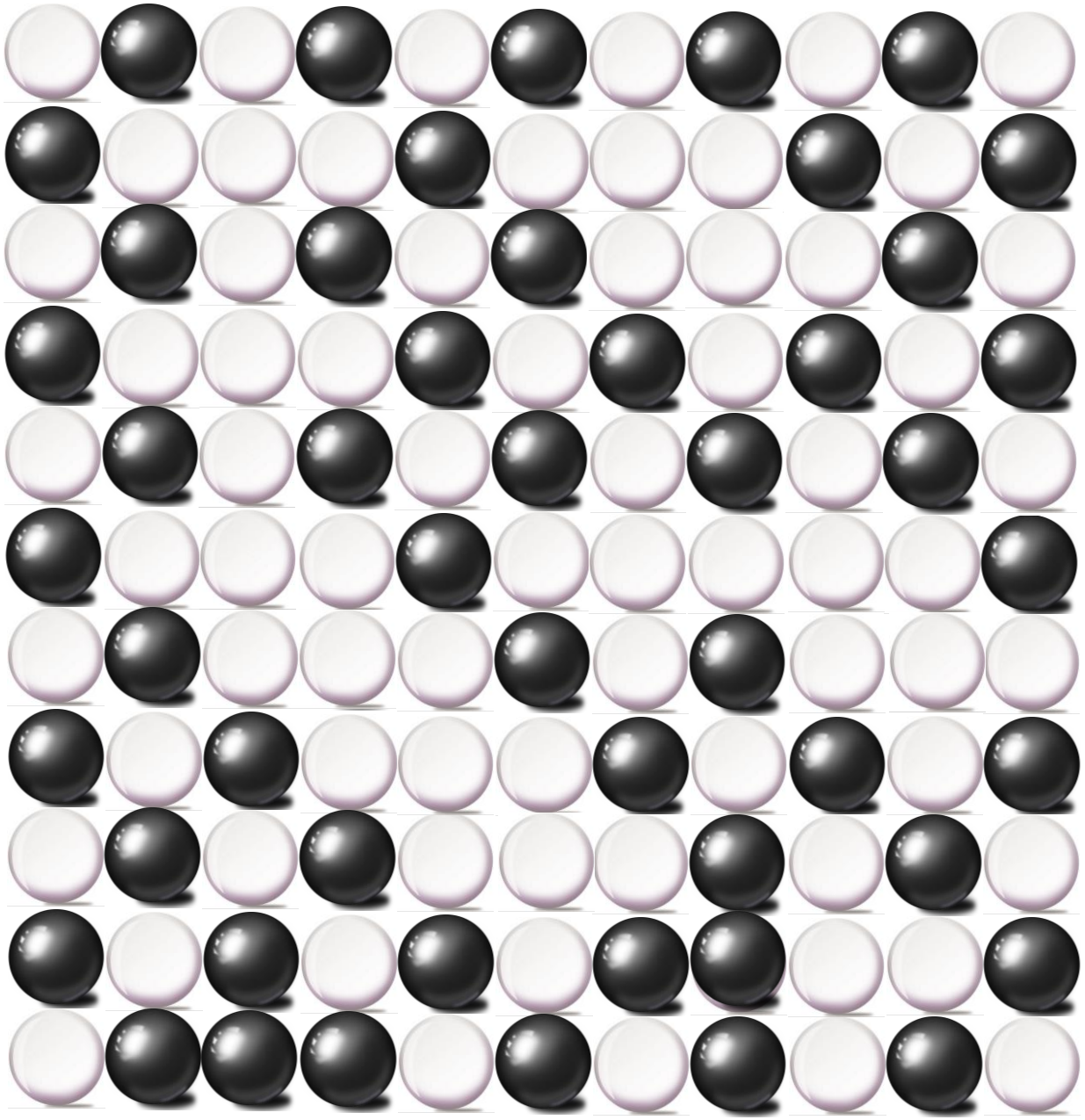
System and complexity

A system is a set of organized, interacting parts which, when complete, exhibits properties or capabilities of the set as a whole which are not attributable exclusively to any of the parts *

- Set
- Organized
- Complete
- Undividable properties
- Interaction
- Holism, hierarchy

A system may be perceived as complex when it does not meet the above criteria

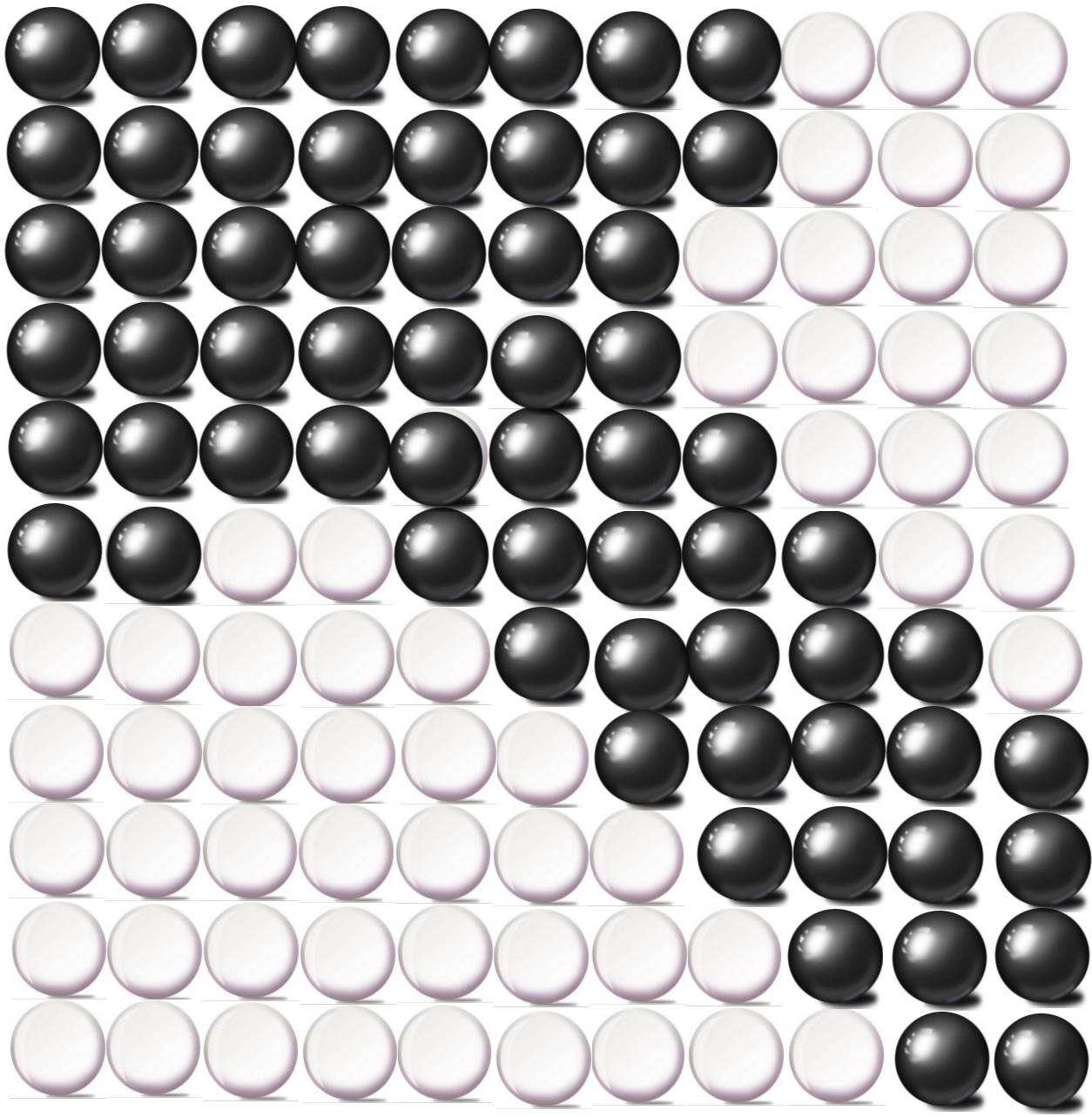
* Derek Hitchins, Getting Grips with Complexity, 2000, Derek Hitchins



Complexity reduced

Getting grips with complexity by

- Defining a set of domain-specific languages,
- each with a single purpose,
- contributing to the properties of the system as a whole



Conclusions & Recommendations

Tooling

- MDE Tooling is not the bottleneck, finding the right language (boundaries) is

Organizational

- Make sure the domain expert is available and keep the dialog open
- We need more system thinkers, and/or…

Process

- …we need to take more time to do proper language design
- We need to keep discipline in splitting-up domains and make them single-aspect
  - Adding states to solve your problem is (temptingly) easy, but adds complexity
- Prevent increase of complexity where possible, e.g. by using cardinality instead of an extra state

Q & A