MathWorks
AUTOMOTIVE
CONFERENCE 2019

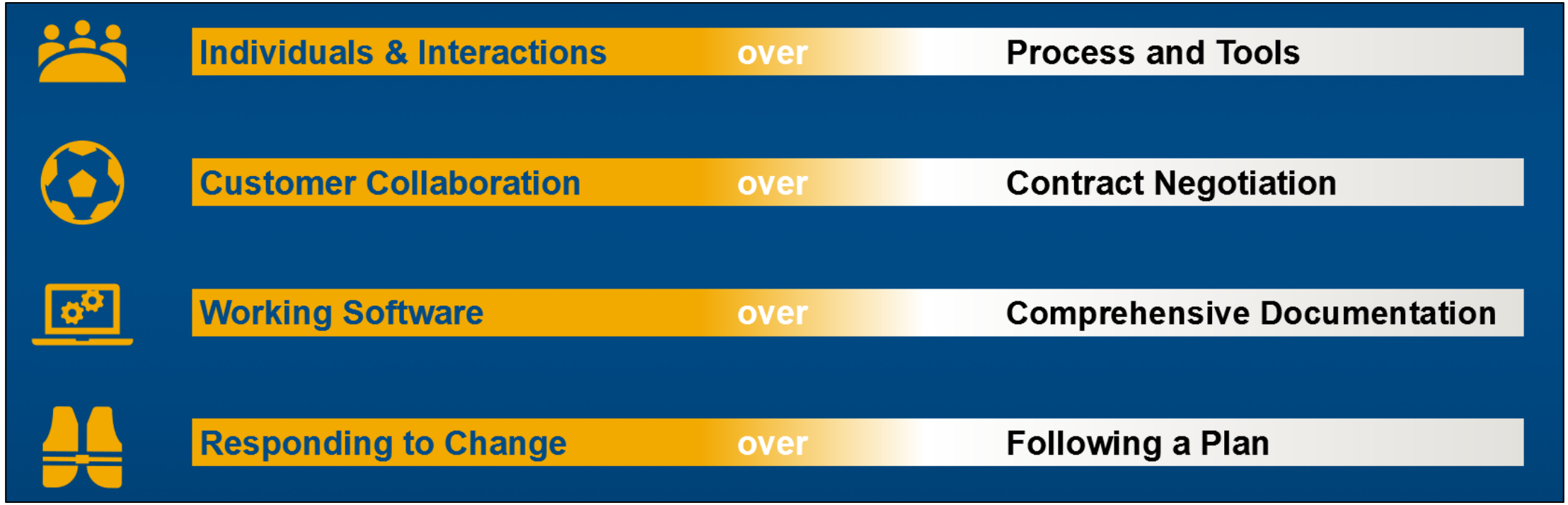
Agile 개발 방법과 모델기반 설계

이영준

Agenda

- Agile Values and Typical Workflow
- Model-Based Design (MBD)
- Agile Development with MBD
- Scrum with MBD

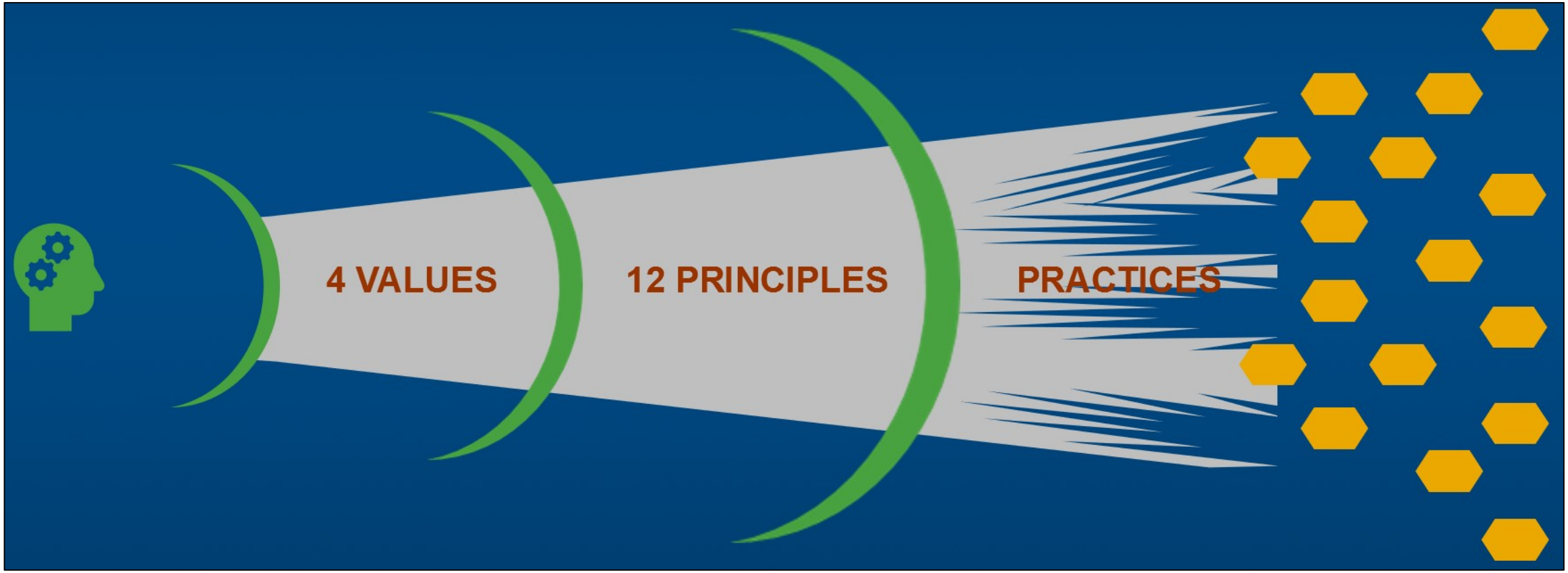
Agile Values



"While there is value in the items on the right, we value the items on the left more."

- The Agile Manifesto, 2001

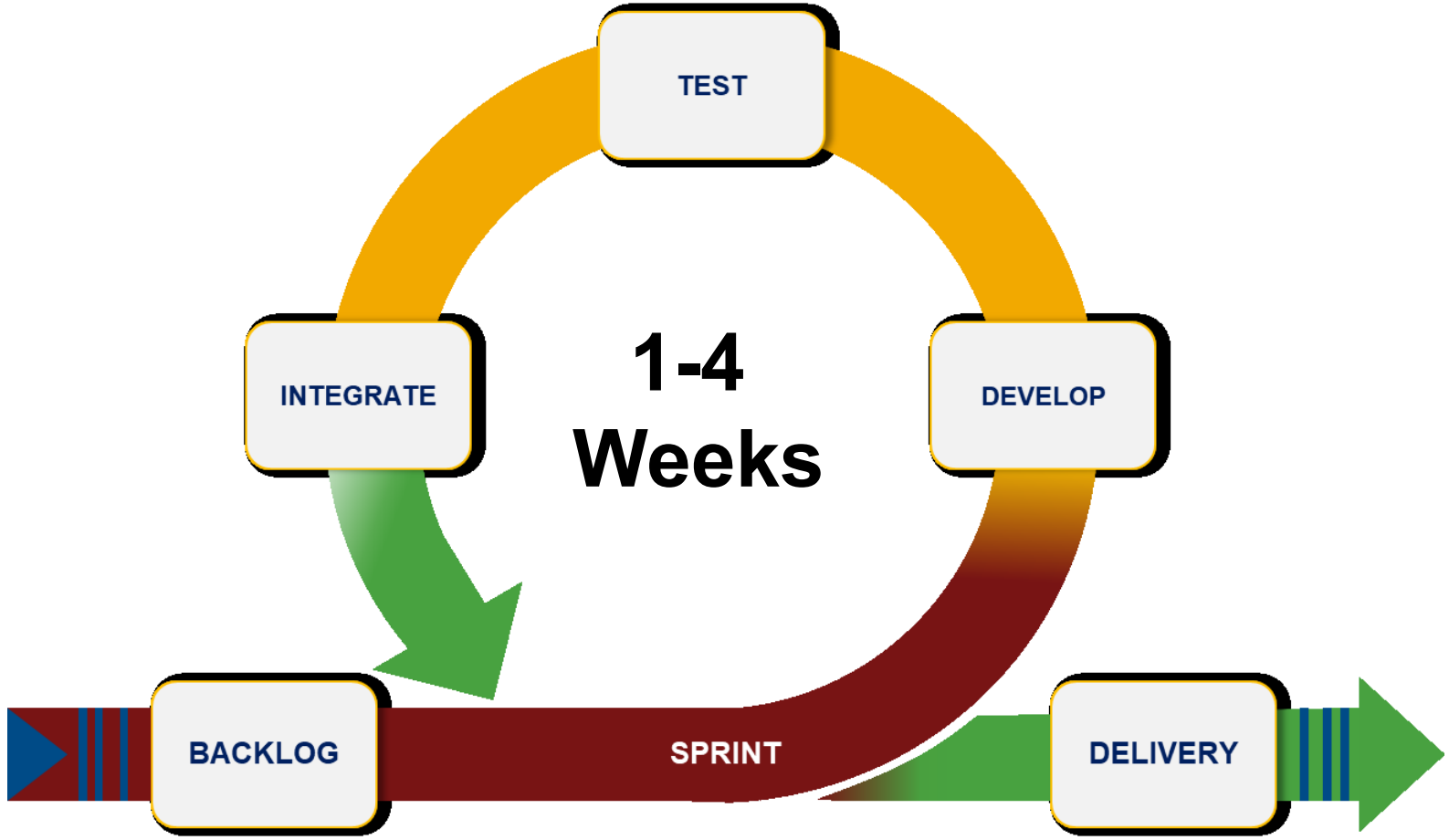
Agile: Values, Principles and Practices



Agile is a mindset defined by values, guided by principles and manifested through many different practices. Agile practitioners select practices based on their needs.

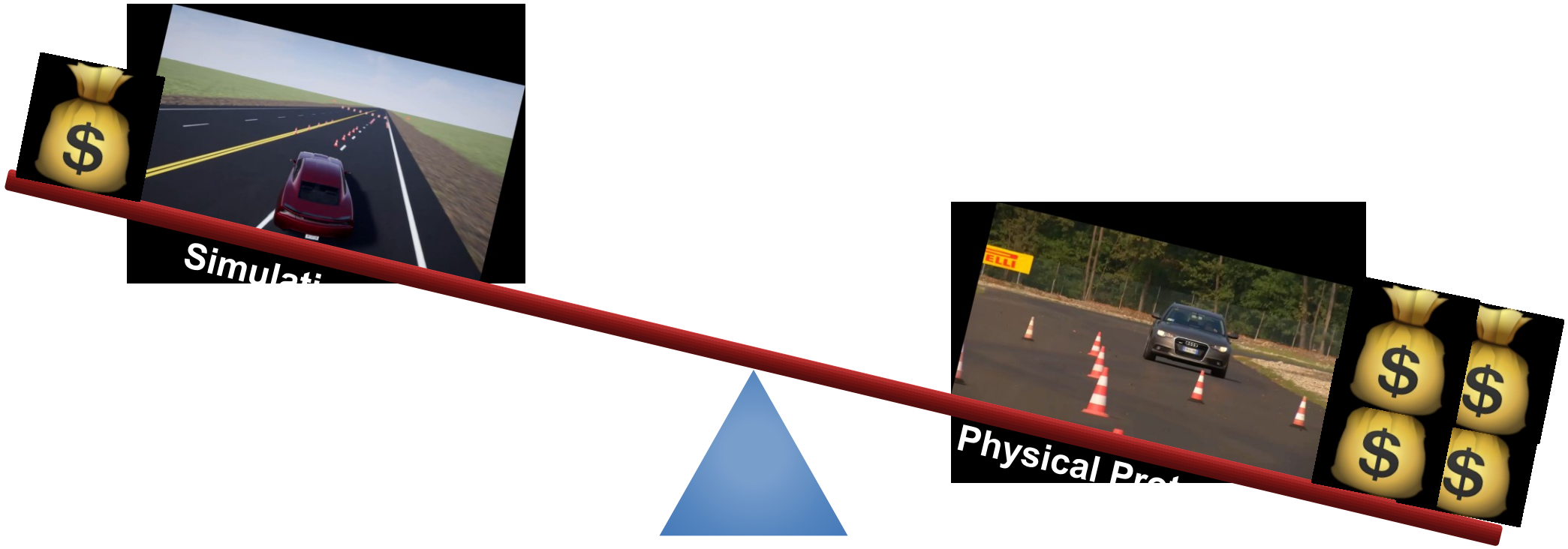
~ Agile Practice Guide (PMI® and Agile Alliance®)

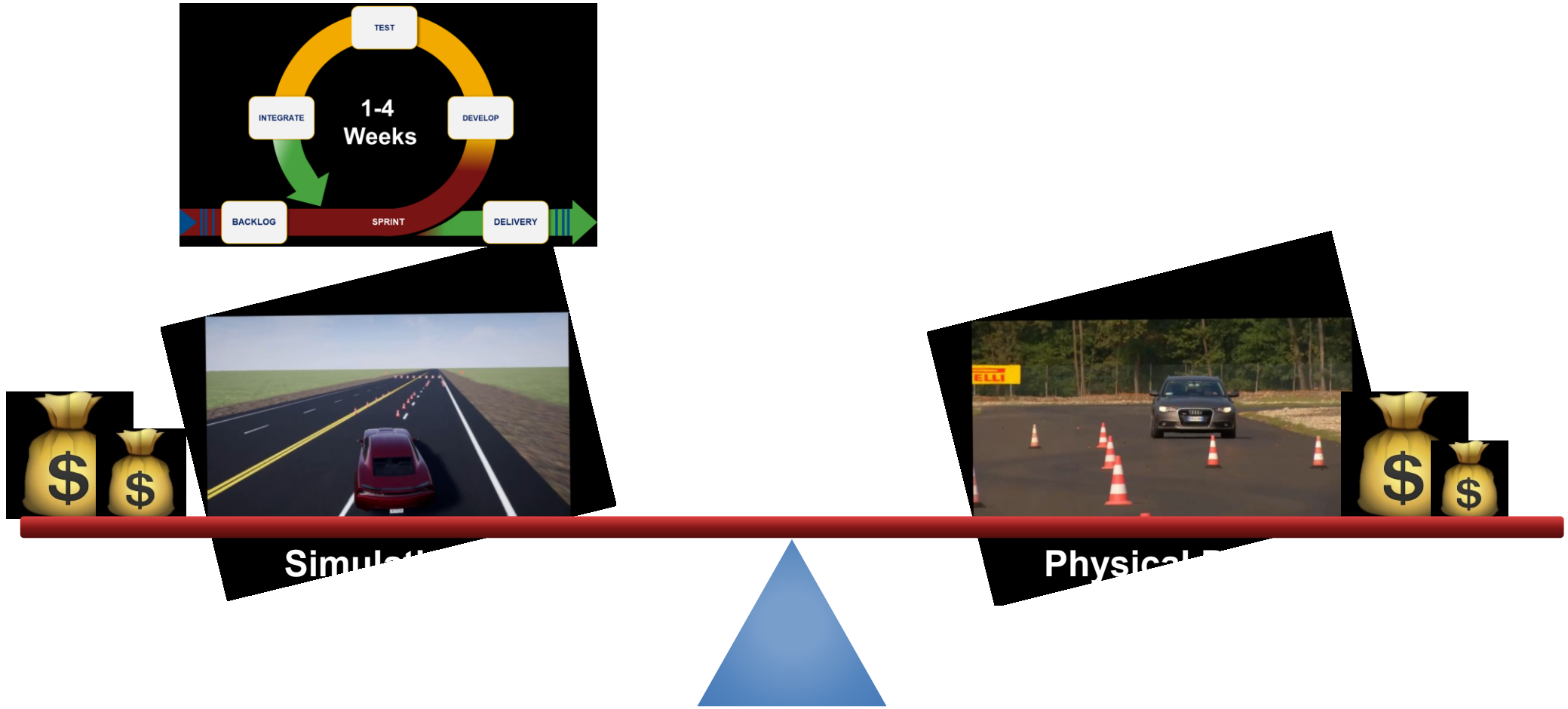
Typical agile development workflow



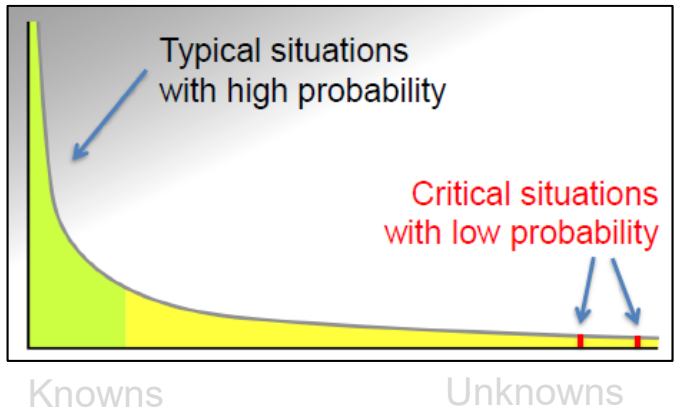
Model-Based Design

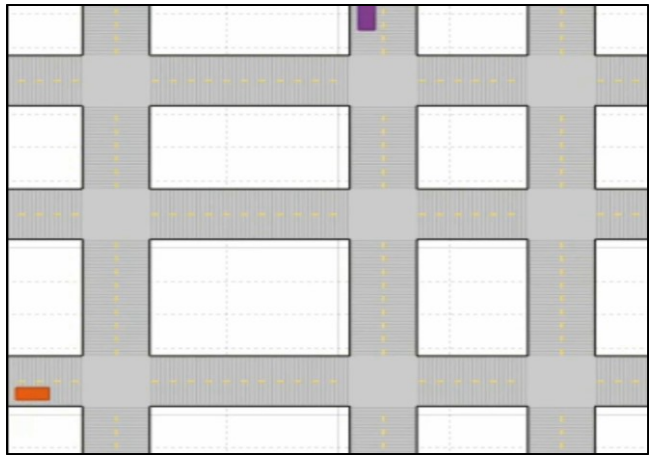
Models == Understanding


$
Simulati
Physical Prot
$
$
$
$


TEST
INTEGRATE
1-4
Weeks
DEVELOP
BACKLOG
SPRINT
DELIVERY

Simulation is key to Level 4-5 autonomy

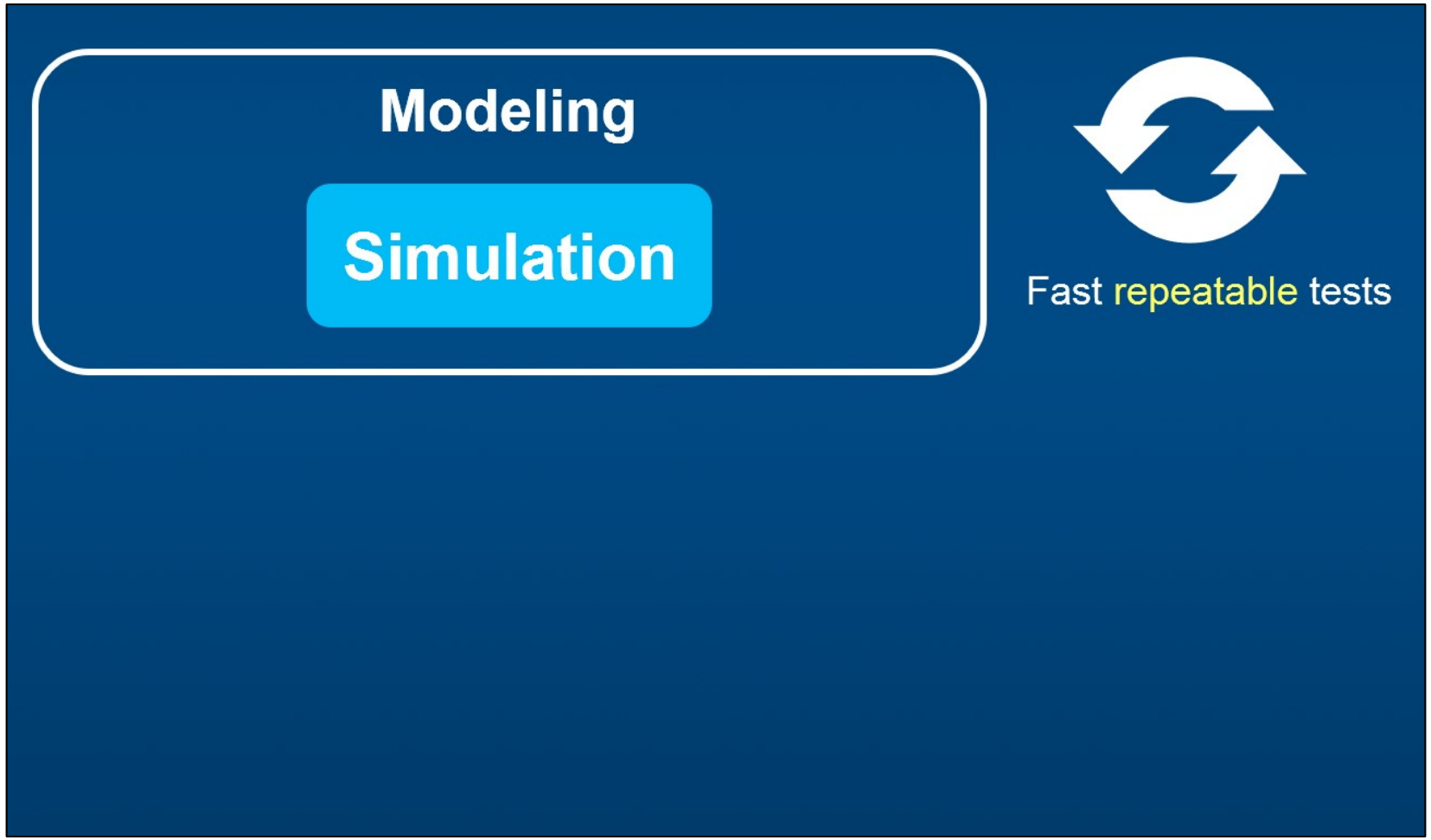




*Source: Center for Artificial Intelligence, Saarland University

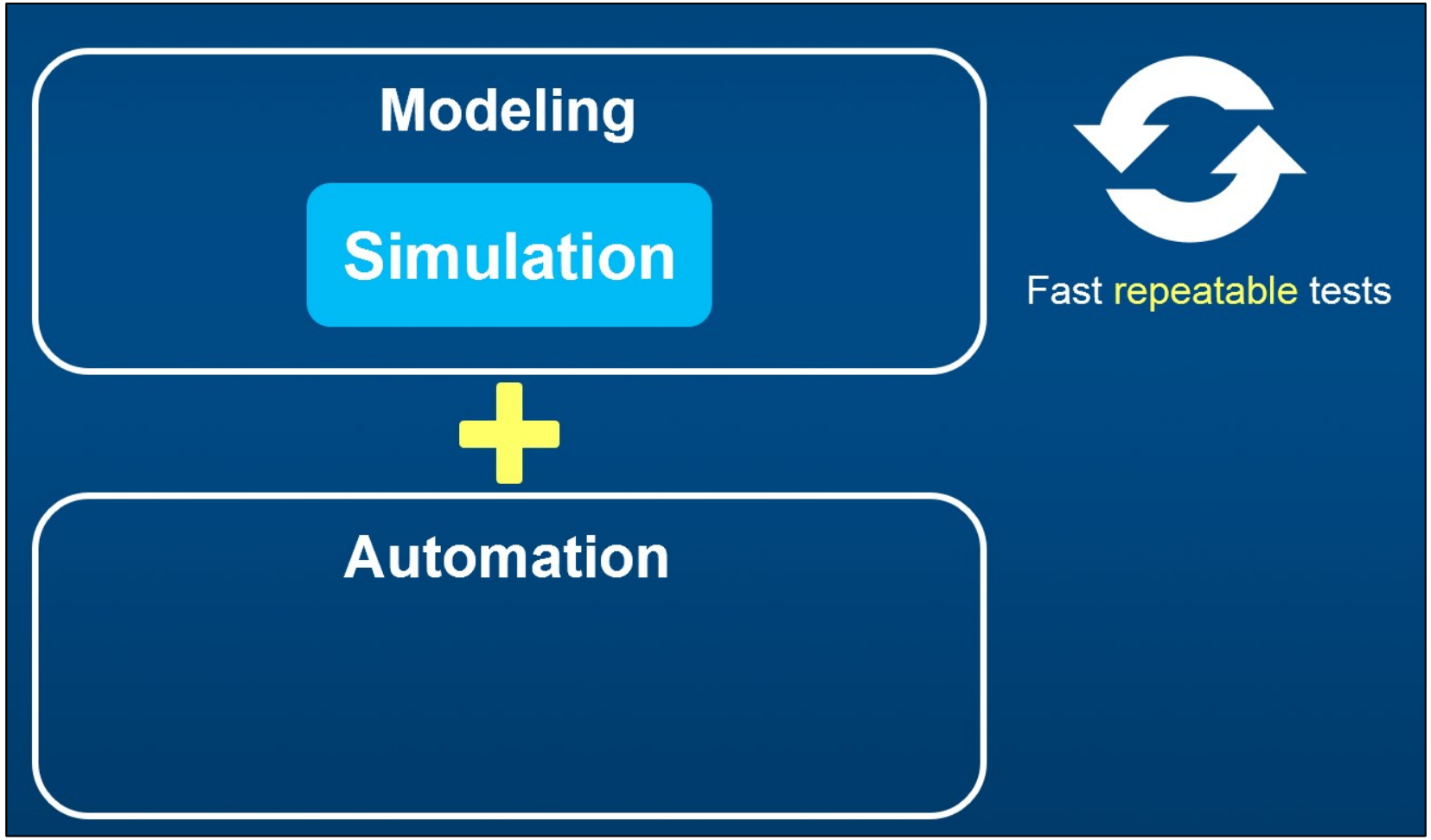
Model-Based Design

Systematic use of **models** **throughout** the development process



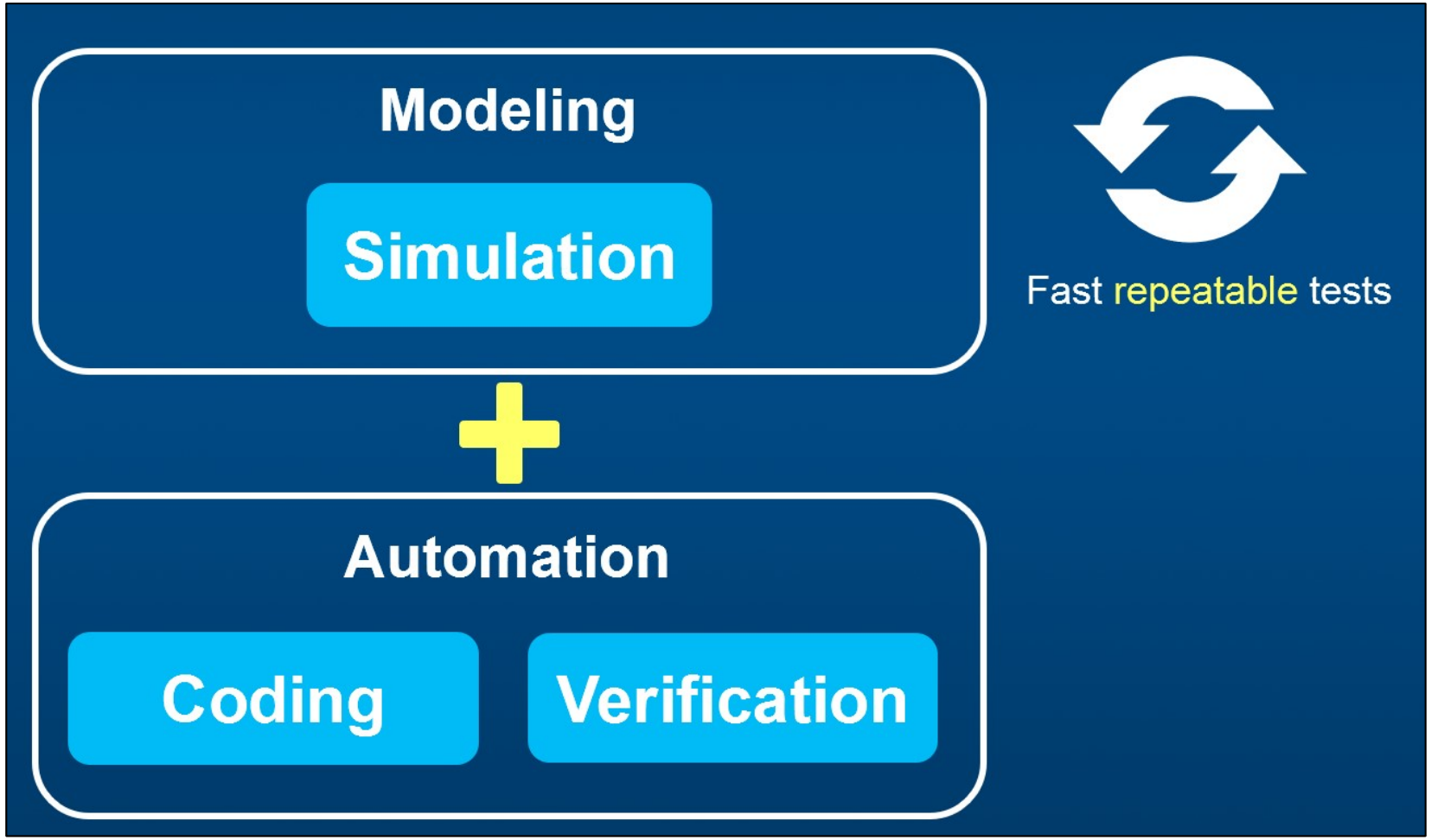
Model-Based Design

Systematic use of **models** **throughout** the development process



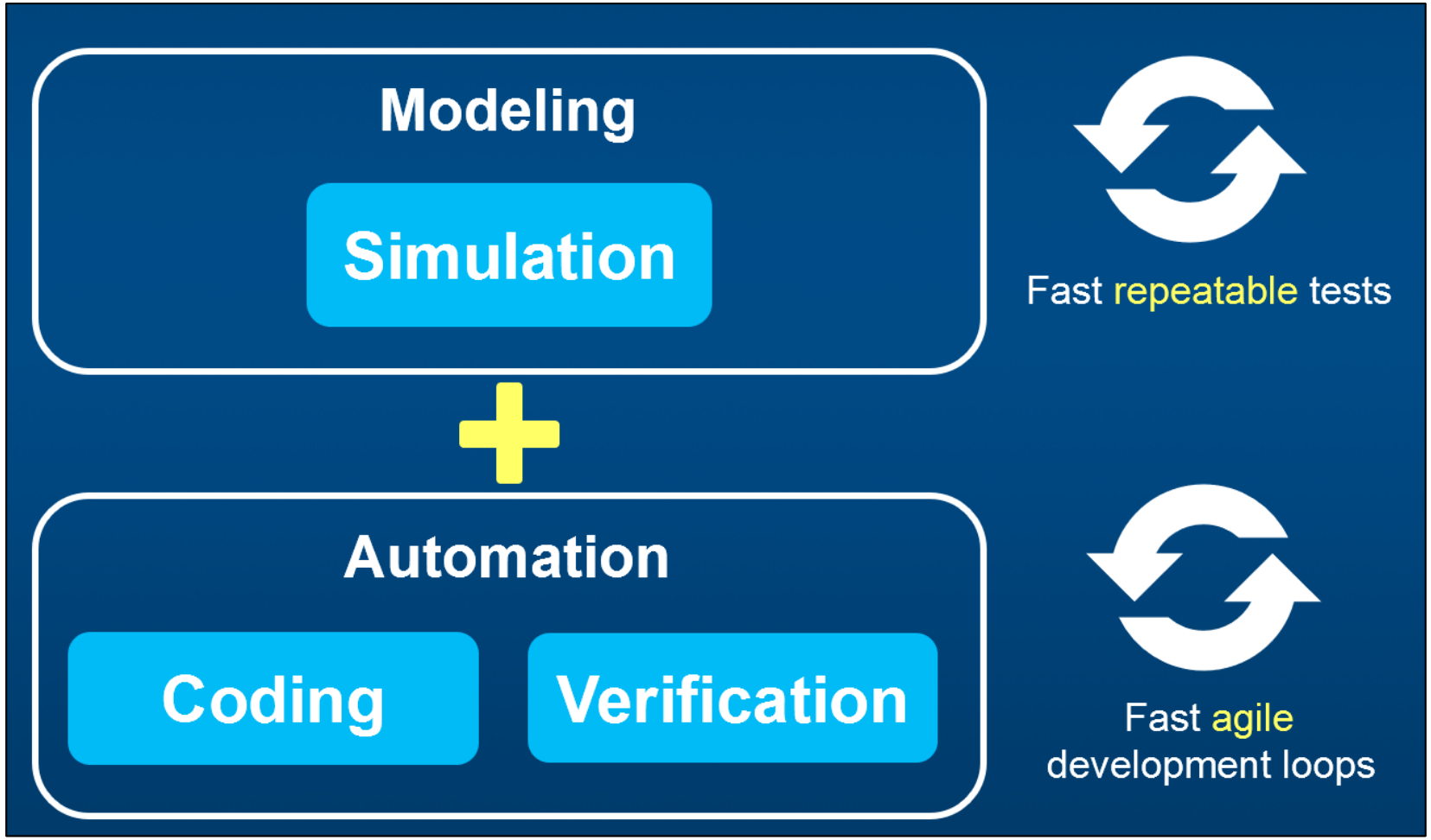
Model-Based Design

Systematic use of **models** **throughout** the development process

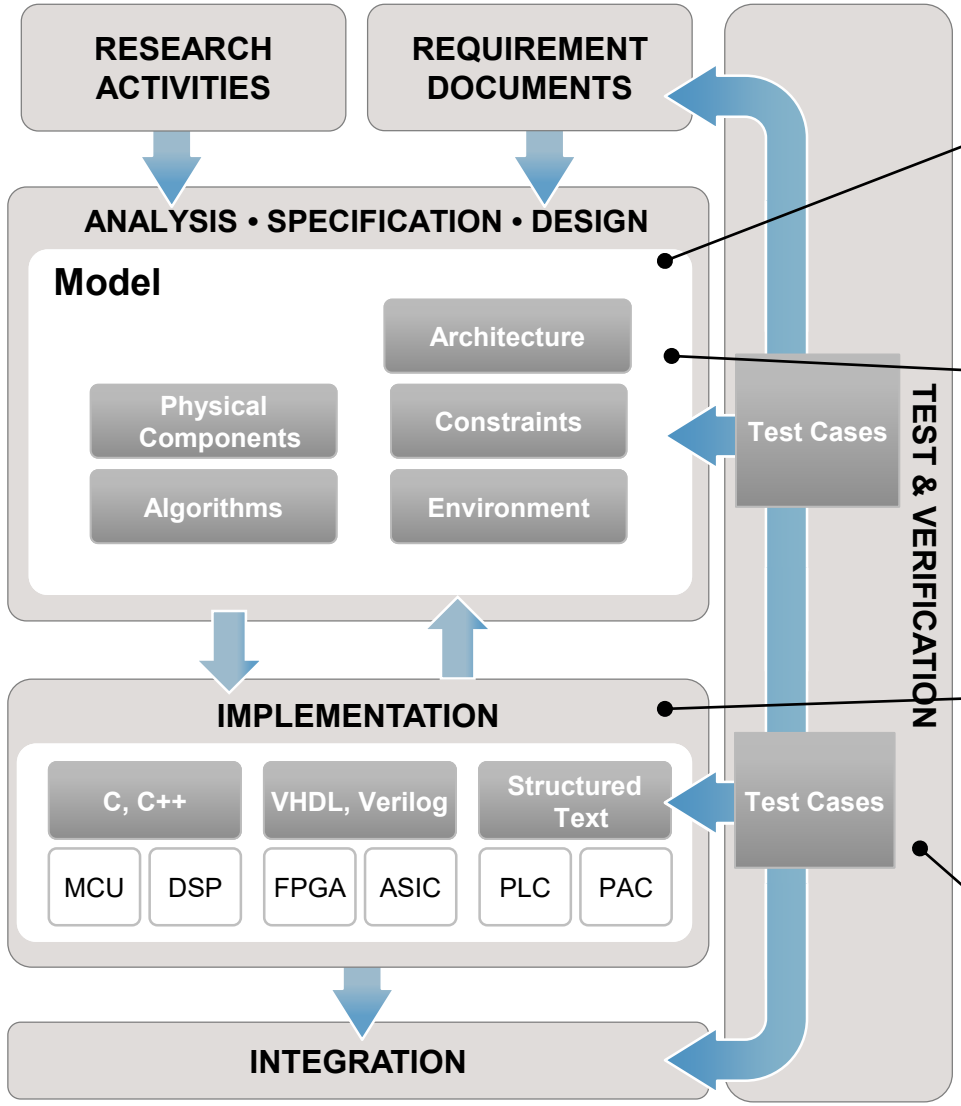


Model-Based Design

Systematic use of models throughout the development process



Model-Based Design



Executable Specification

- Unambiguous – easy to understand
- Systems engineering - modeling whole system including environment
- Sharing of models to improve communication and collaboration
- Early validation and test development

Multi-domain Design

- Model algorithms and environment
- Perform integration testing at model level before implementation

Automatic Code Generation

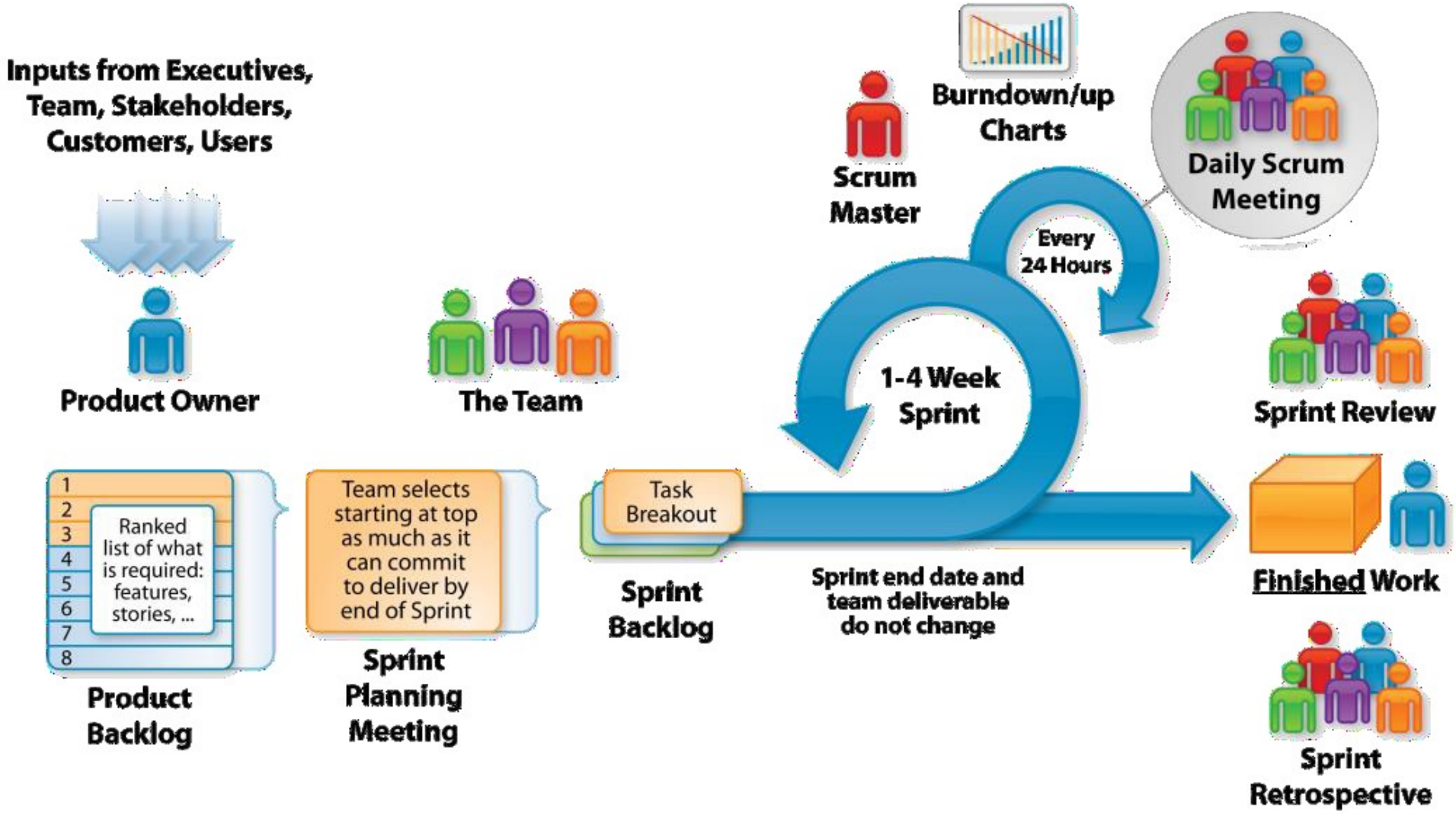
- Eliminate errors from hand-coding
- Regenerate easily for different targets

Continuous Test and Verification

- Detect errors early in development
- Reduce use of physical prototypes
- Reuse tests throughout development process

Agile Development with MBD

Agile Development – Key Principles

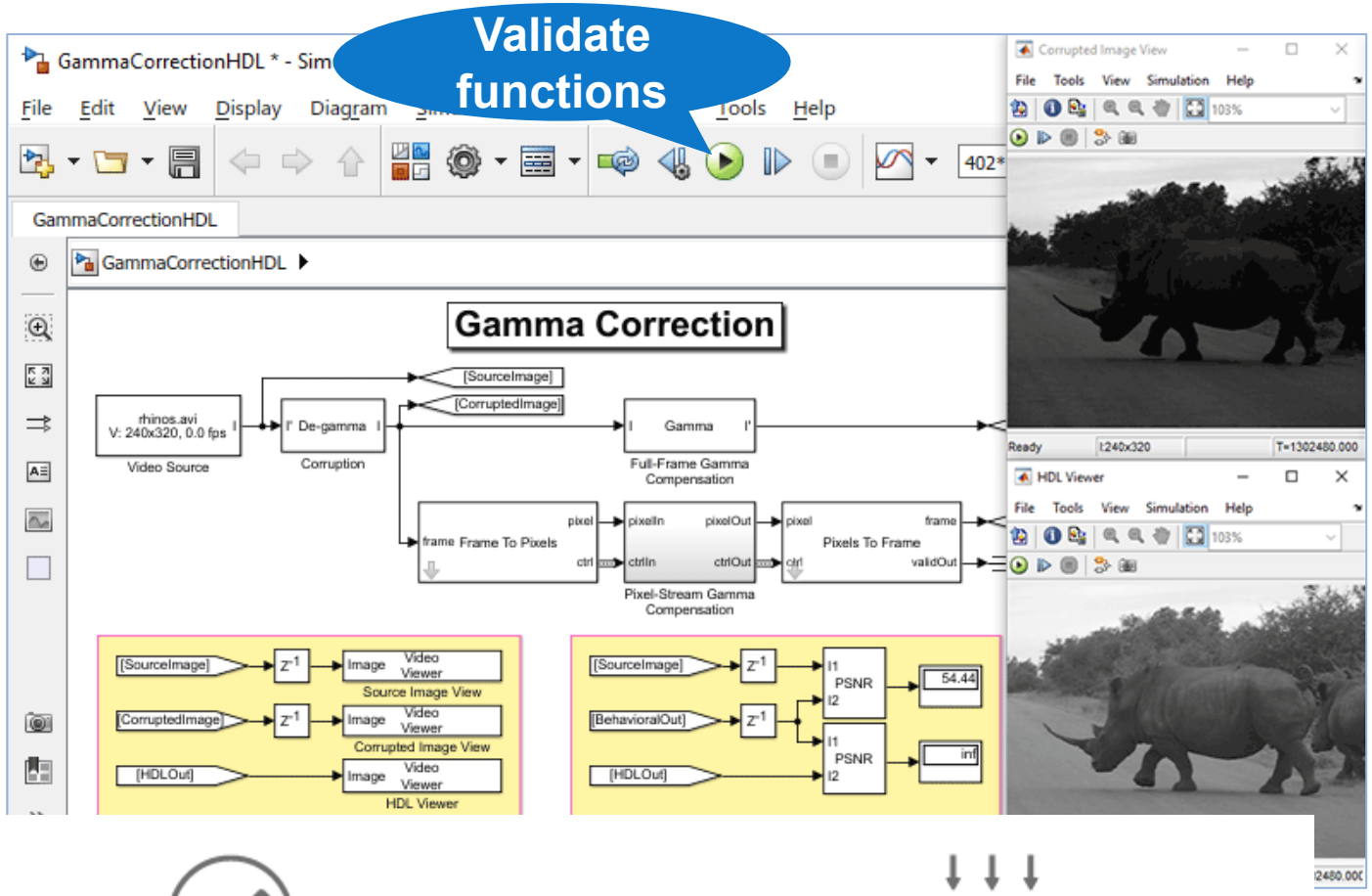


Customer satisfaction by **rapid delivery of useful software**

Picture Source: http://www.softwareplant.com/scrum/

Principle: Customer satisfaction by rapid delivery of useful software

- **Simulation** allows customer evaluation of functional behavior **early and often**.
- **Useful software** can be delivered throughout the project via **code generation**.

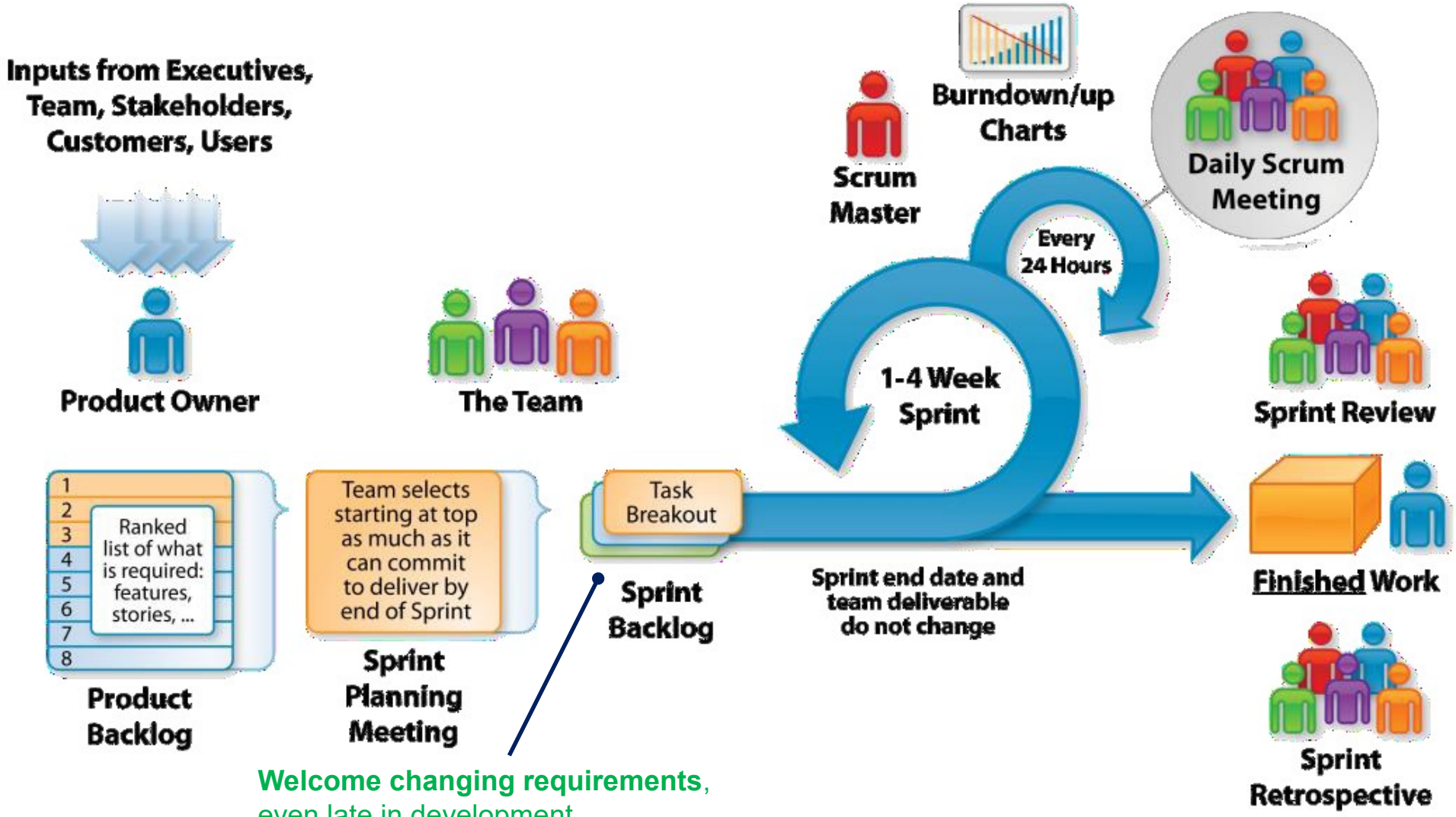


Model and Simulate Your System

Test Early and Often

Automatically Generate Code

Agile Development – Key Principles



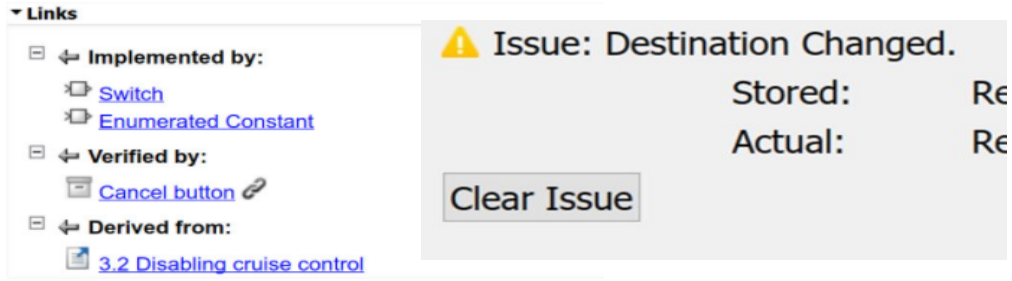
Welcome changing requirements, even late in development

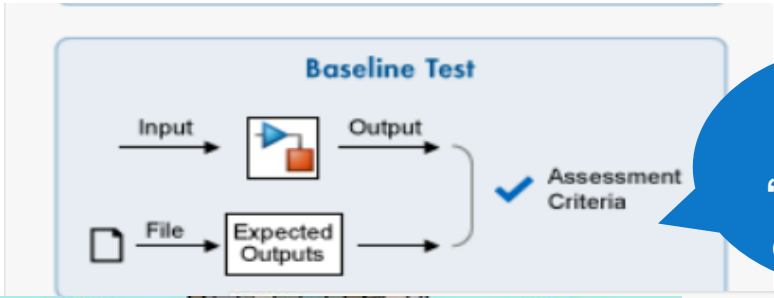
Customer satisfaction by **rapid delivery of useful software**

Picture Source: http://www.softwareplant.com/scrum/

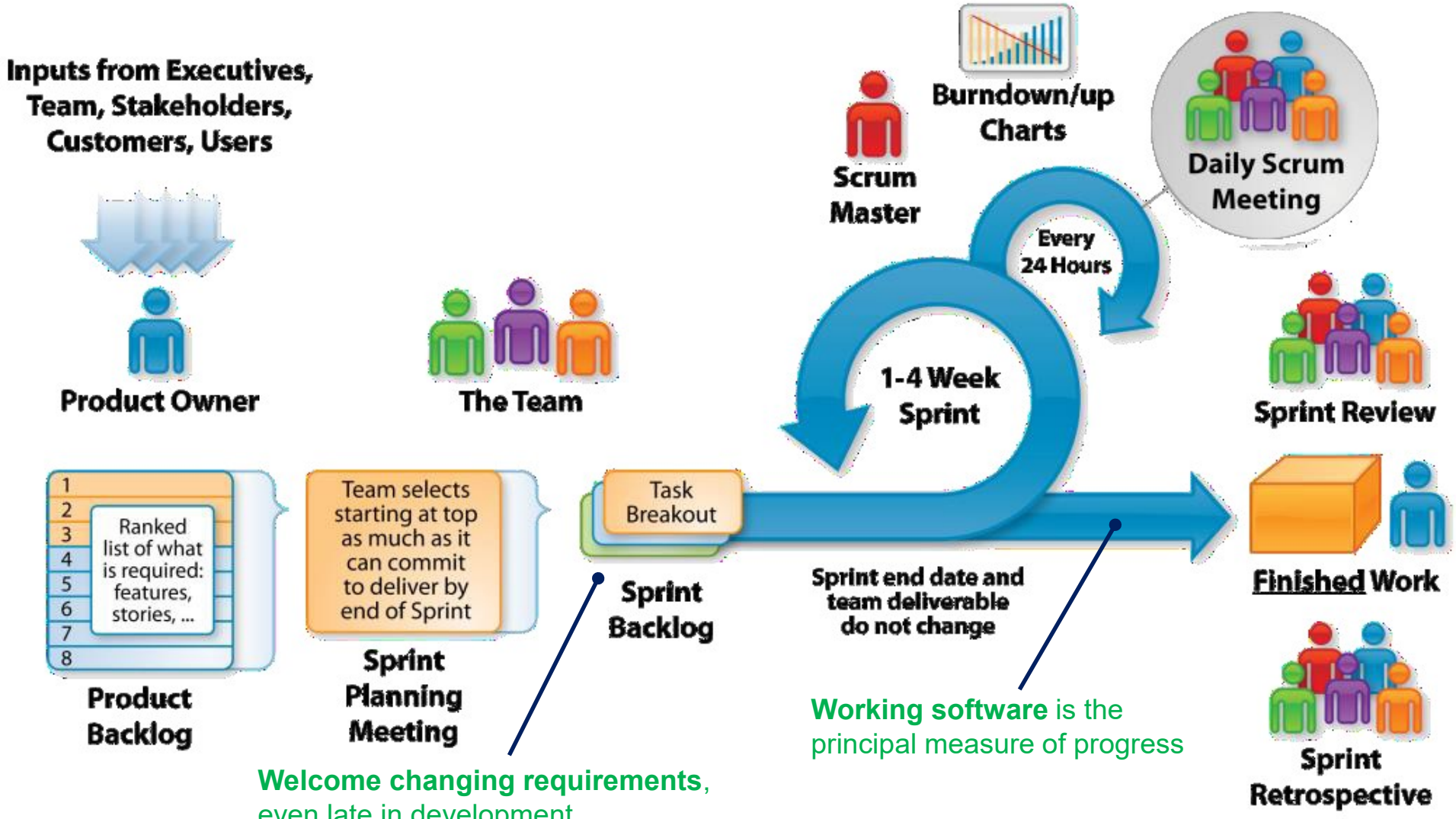
Principle: Welcome changing requirements, even late in development

- **Requirements traceability** supports **impact analysis** of affected model components.
- Dynamic/executable models allow for **rapid evaluation of requirements** changes.
- **Regression testing** of simulation-based tests can be automated to confirm **new design meets existing requirements**.





Agile Development – Key Principles

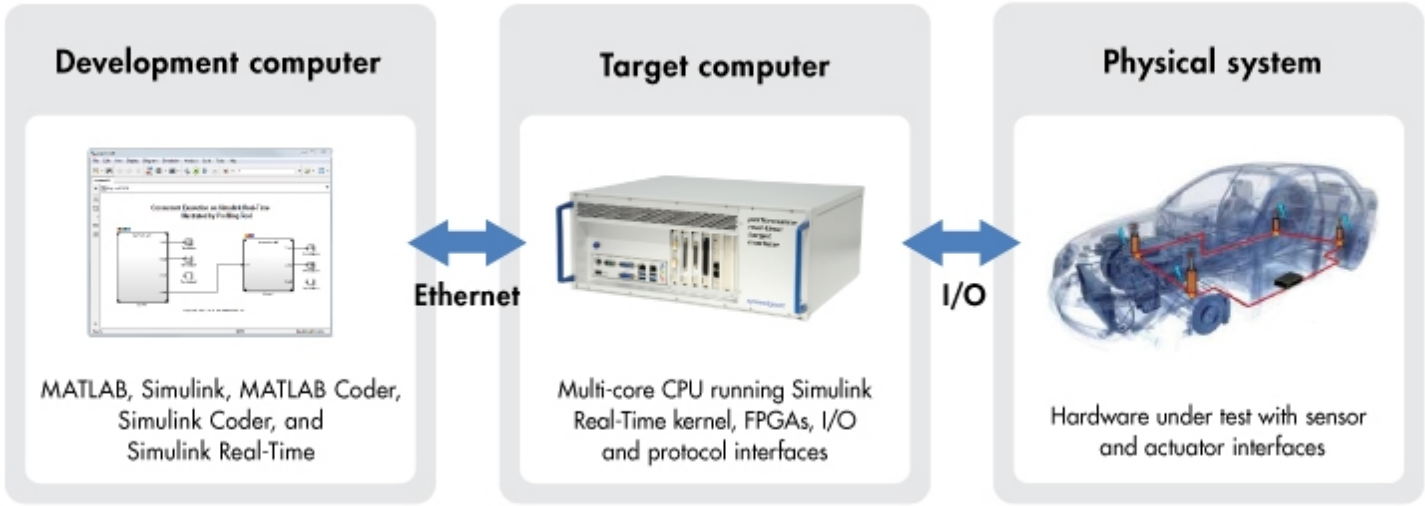


Customer satisfaction by **rapid delivery of useful software**

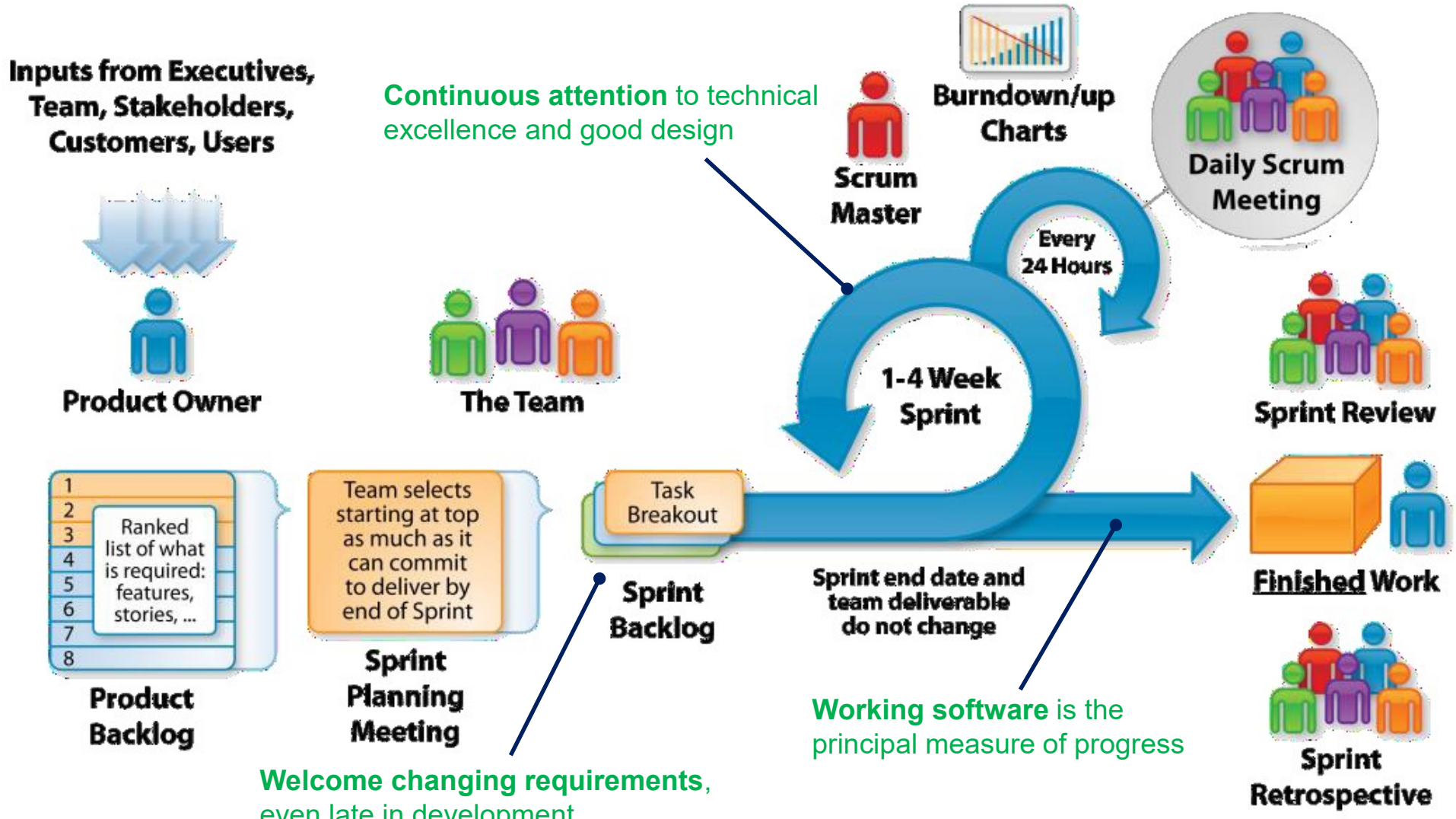
Picture Source: http://www.softwareplant.com/scrum/

Principle: Working software is the principal measure of progress

- Functional designs can be evaluated continuously via **executable models**.
- Code generation supports rapid software deliveries, **rapid prototyping** and **HIL test**



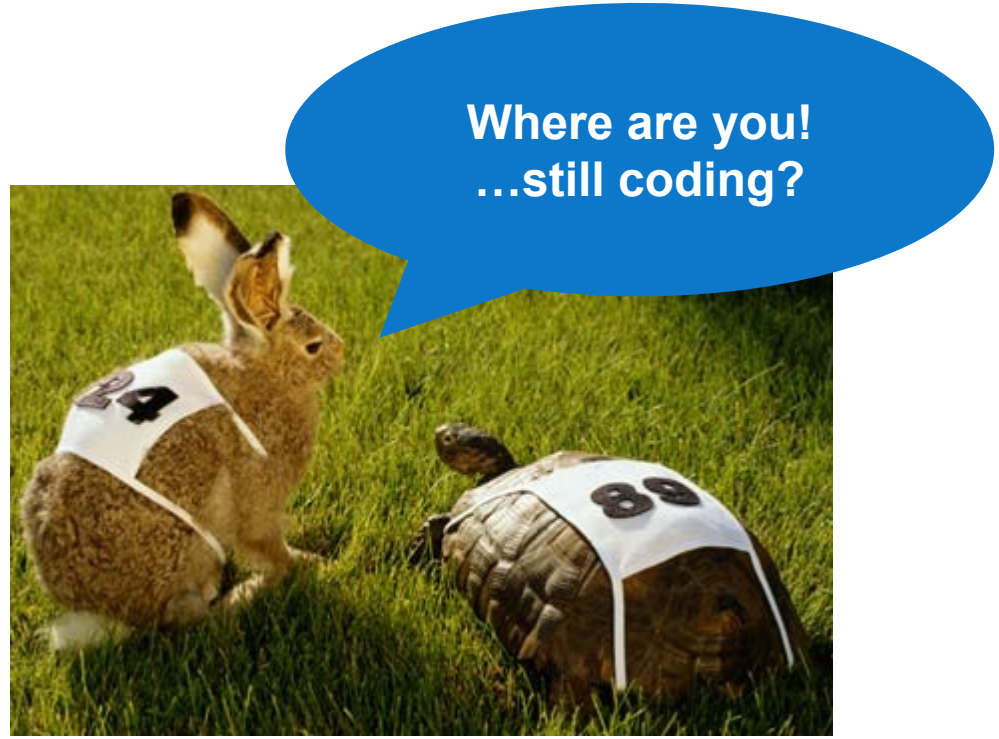
Agile Development – Key Principles

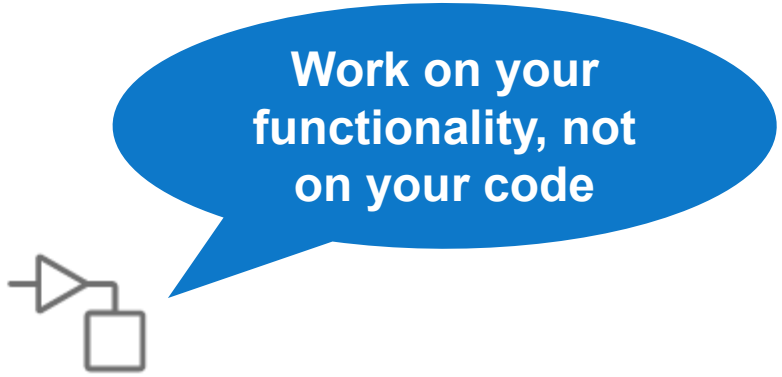


Picture Source: http://www.softwareplant.com/scrum/

Principle: Continuous attention to technical excellence and good design

- **Continuous development and testing** in an executable modeling environment allows for – "**build a little / test a little**" workflows.



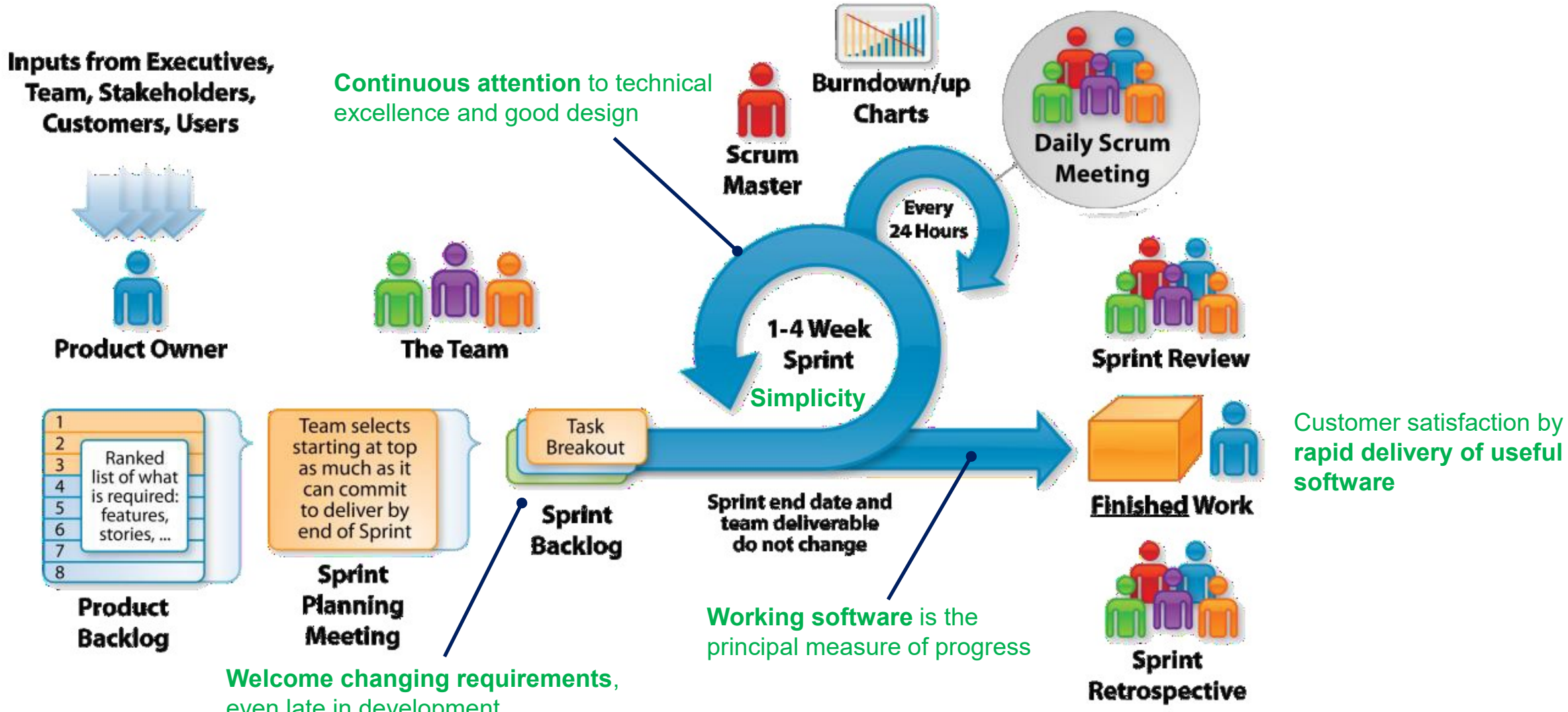


Model and Simulate Your System

Test Early and Often

Automatically Generate Code

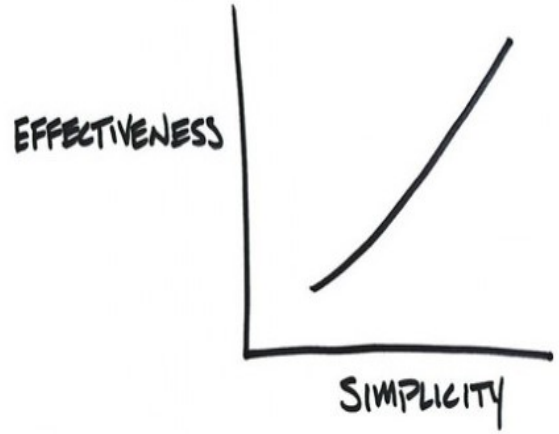
Agile Development – Key Principles



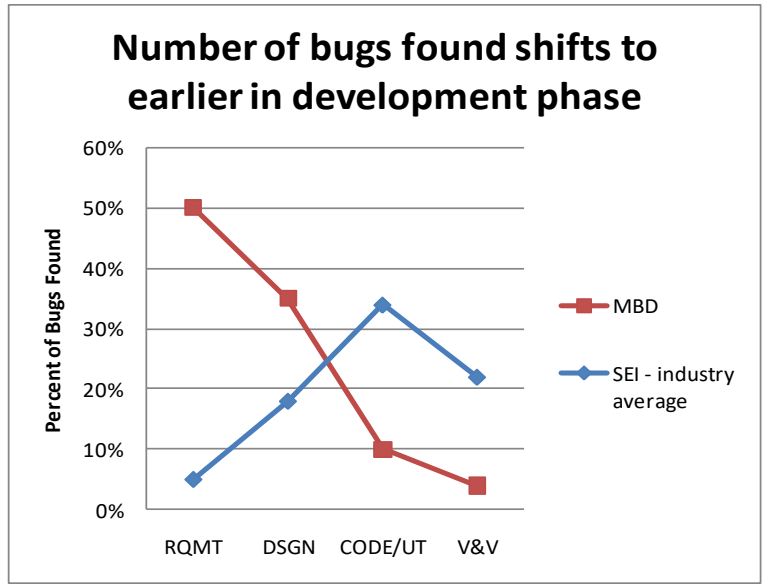
Picture Source: http://www.softwareplant.com/scrum/

Principle: Simplicity—the art of maximizing the amount of work not done—is essential

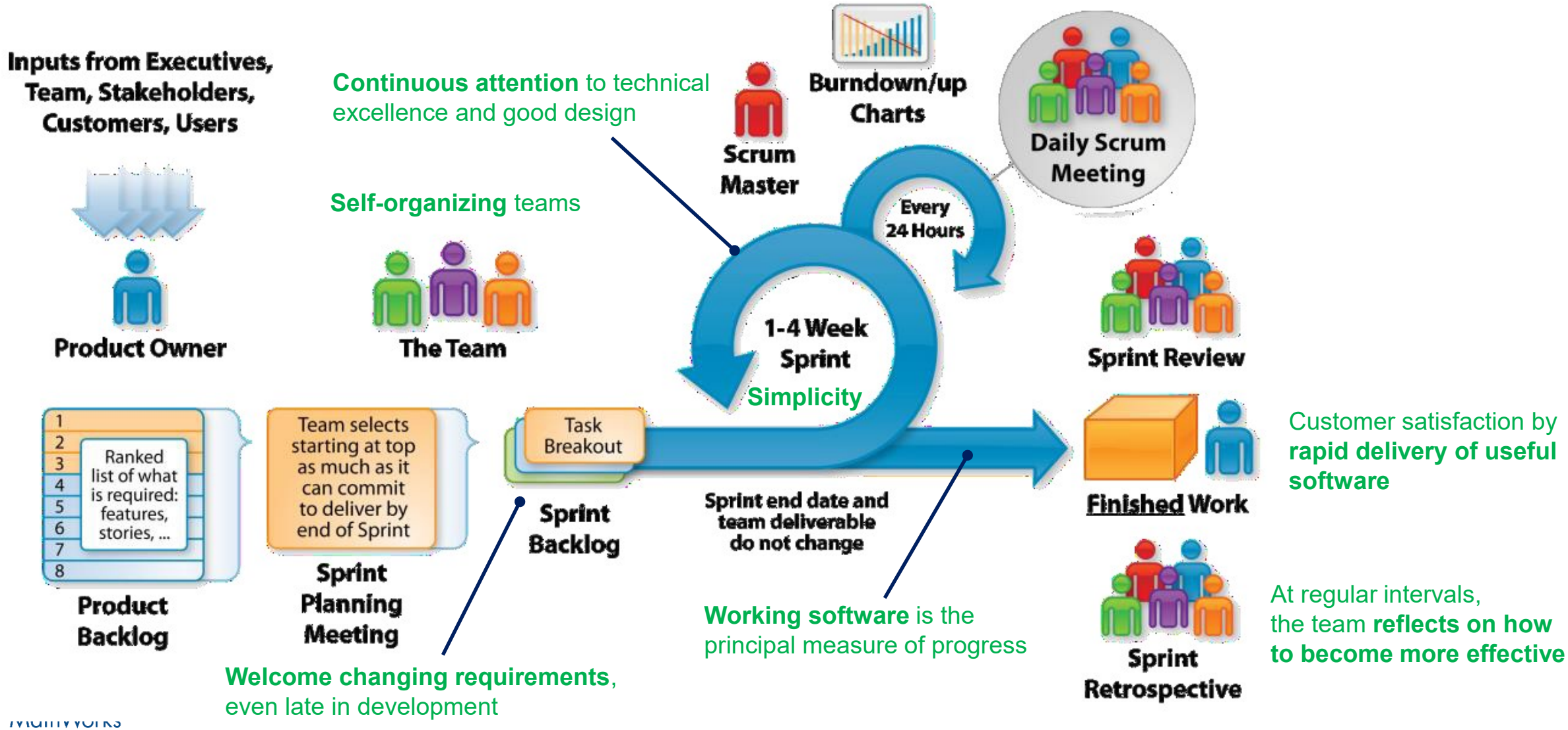
- Model-Based Design supports identification of bugs where they are introduced and “cheaper” to fix.



- Reduces rework (design and test).



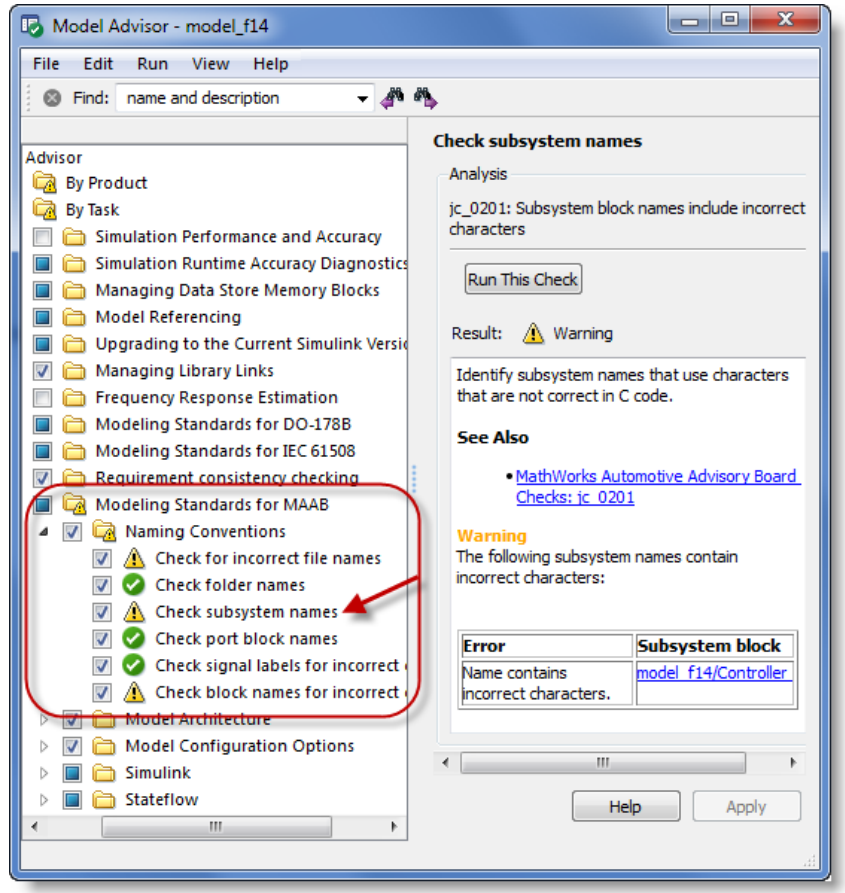
Agile Development – Key Principles



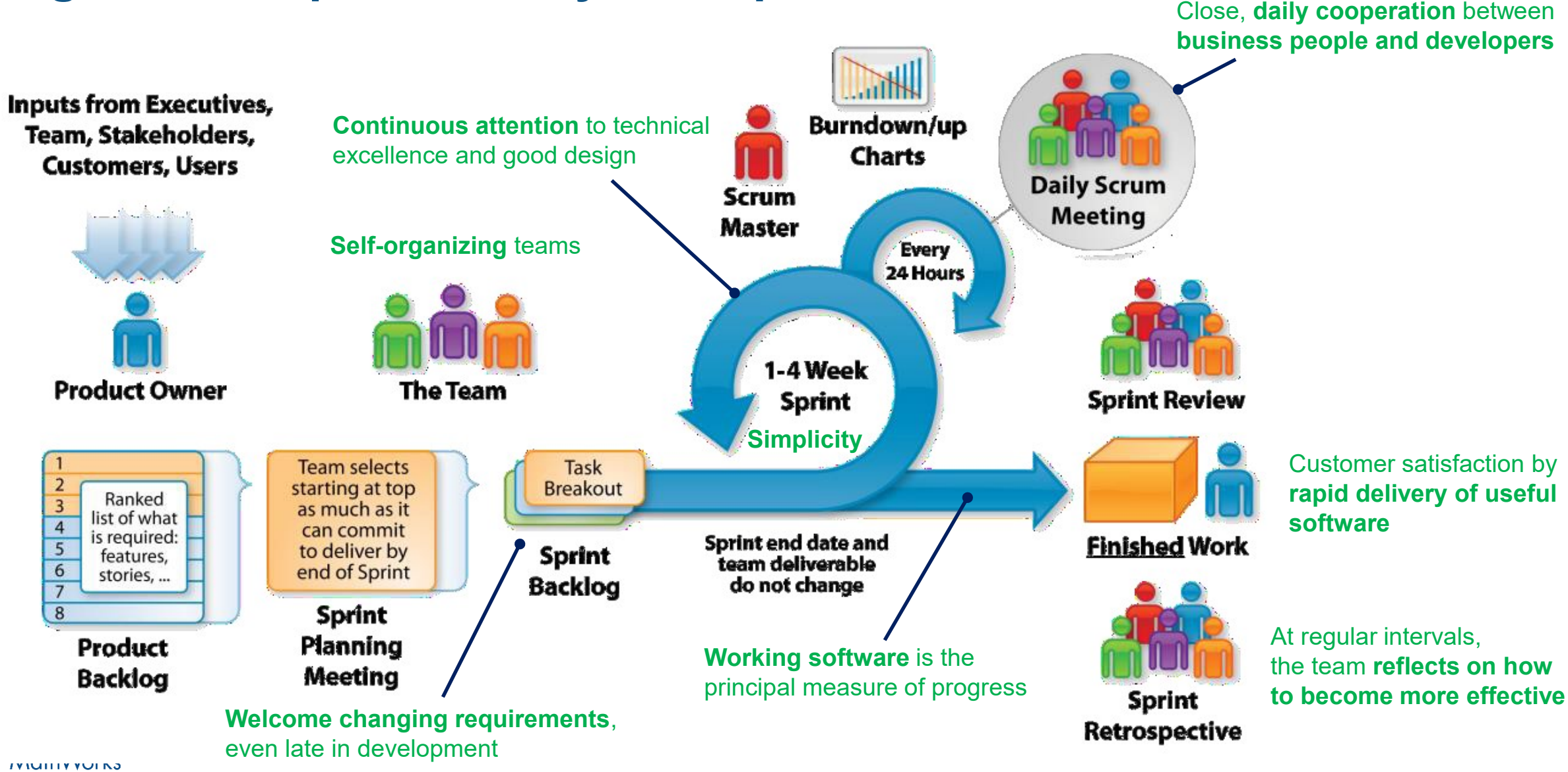
Picture Source: http://www.softwareplant.com/scrum/

Principle: At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly

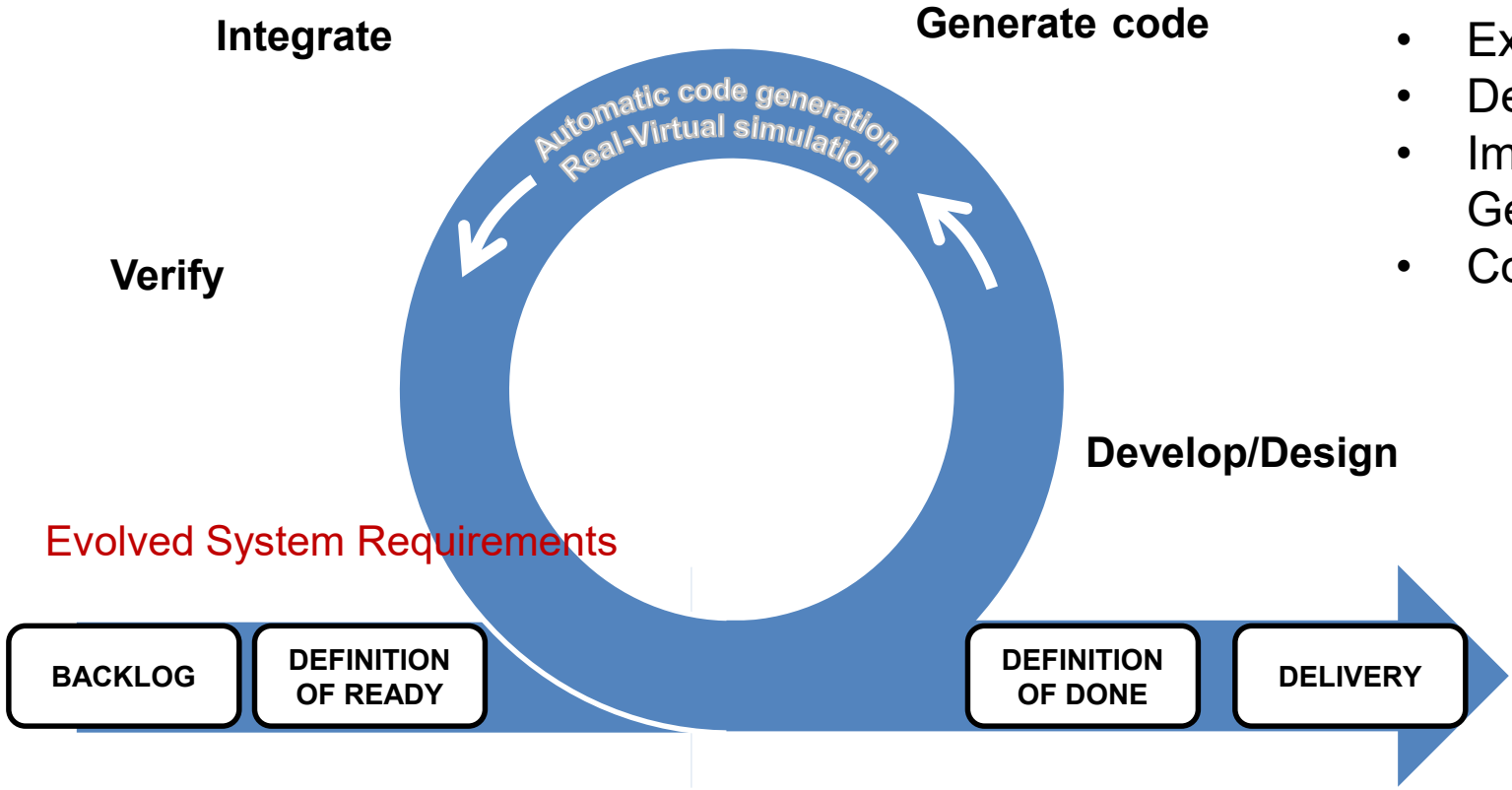
- **Collect metrics from models** to measure and improve process e.g. test coverage, model metrics
- Flexible tool suite supports wide variety of workflows and development processes.
- Open APIs.



Agile Development – Key Principles



Scrum with MBD



Model-Based Design

- Executable Specifications from Models
- Design with Simulation
- Implementation with Automatic Code Generation
- Continuous Test and Verification

Develop

Integrate

Generate code

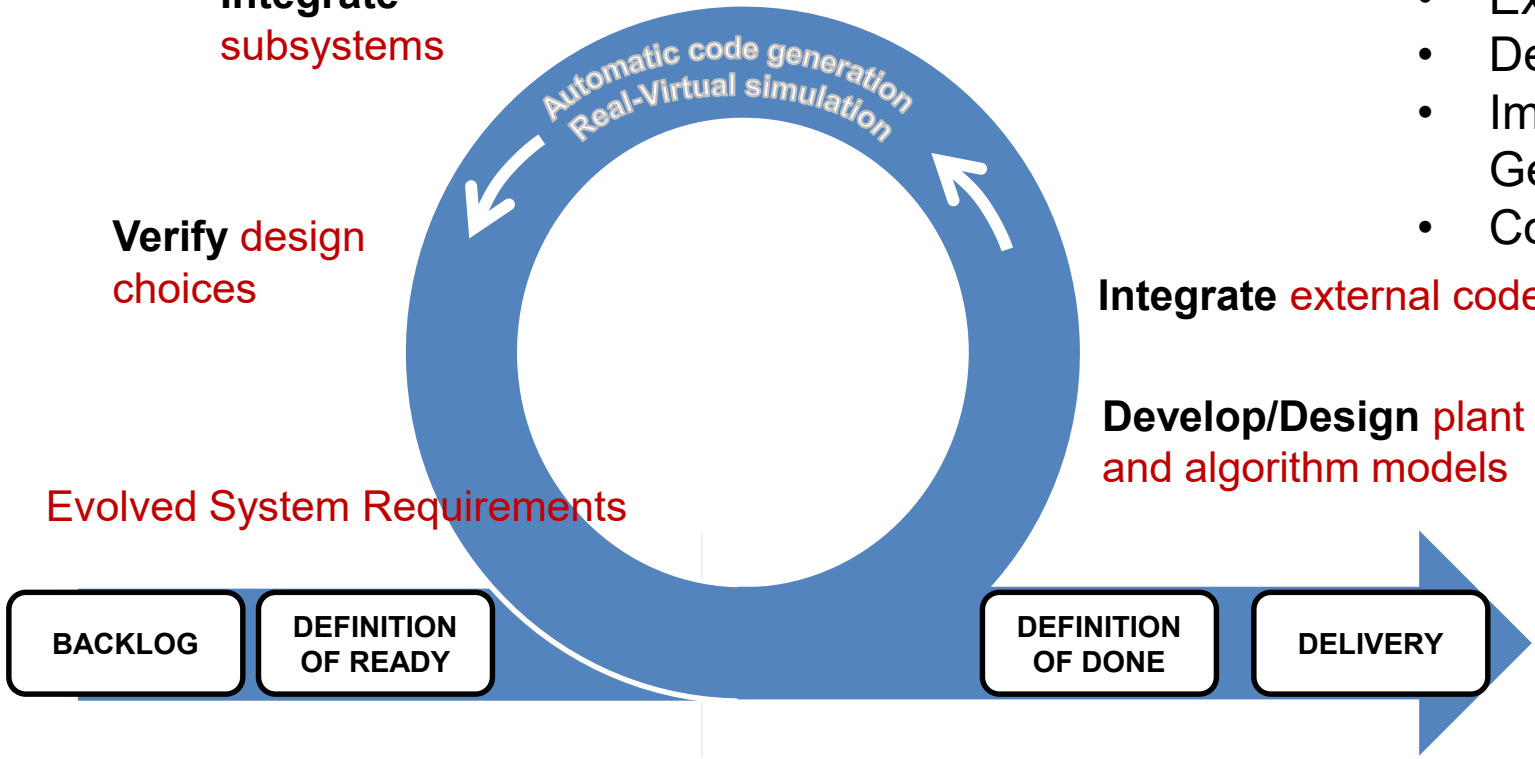
Simulate

Test

MIL – Modeling only

Integrate subsystems

Verify design choices



Evolved System Requirements

Integrate external code

Develop/Design plant and algorithm models

Model-Based Design

- Executable Specifications from Models
- Design with Simulation
- Implementation with Automatic Code Generation
- Continuous Test and Verification

Test

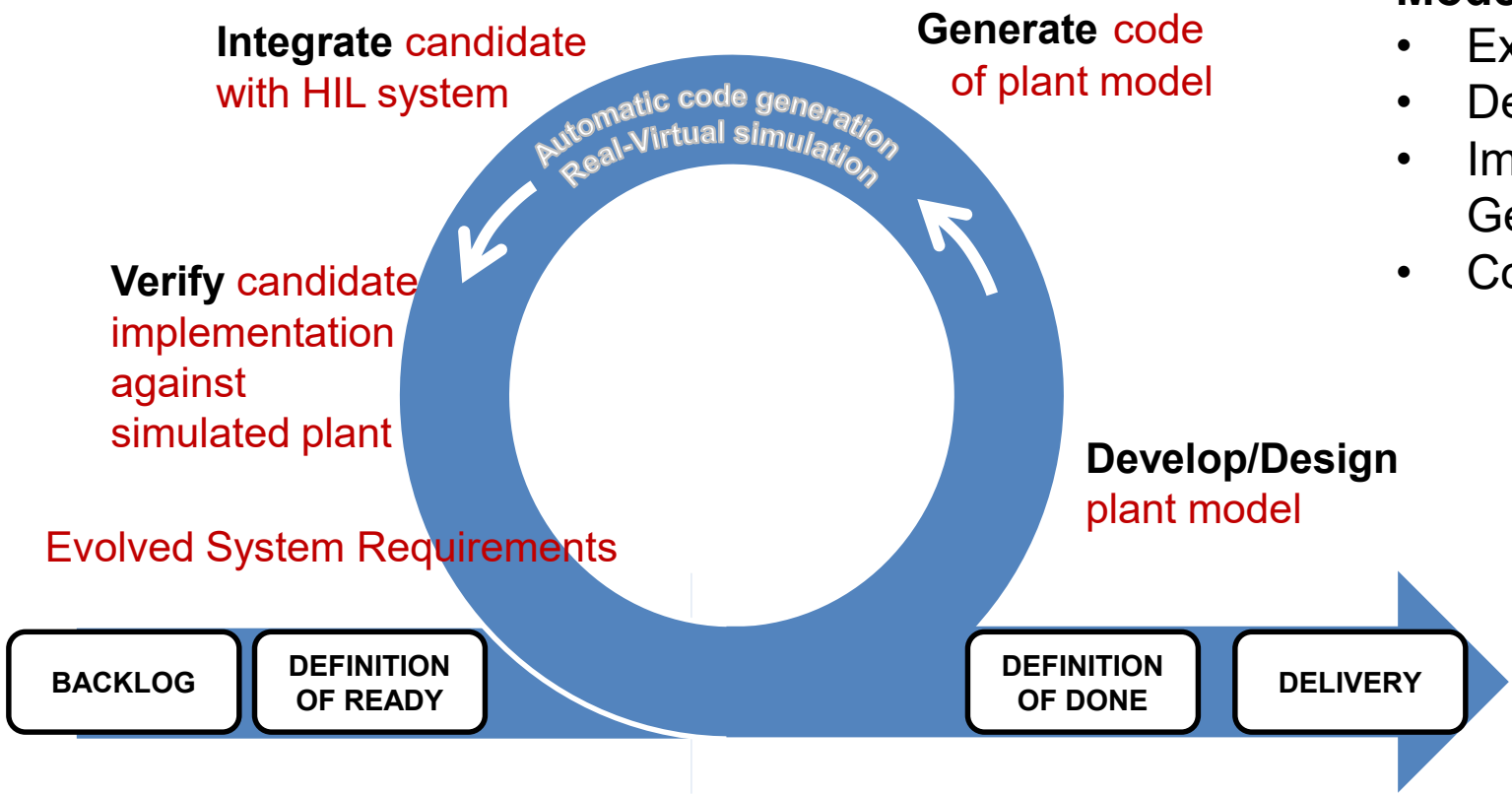
Simulate

Generate code

Evaluate/Decide

Validation

HIL Verification



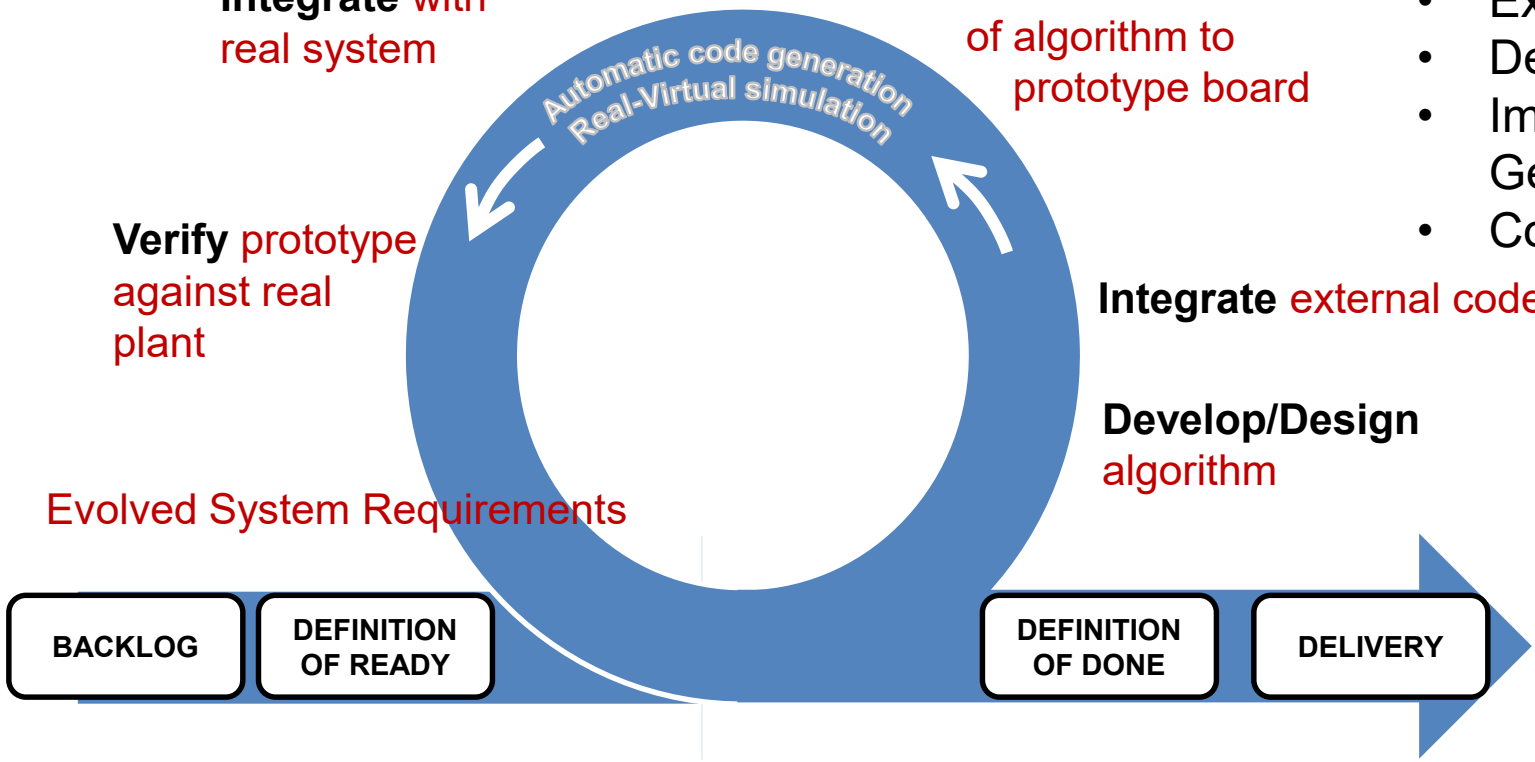
Model-Based Design

- Executable Specifications from Models
- Design with Simulation
- Implementation with Automatic Code Generation
- Continuous Test and Verification

Integrate external code

Simulate Test

Rapid prototyping

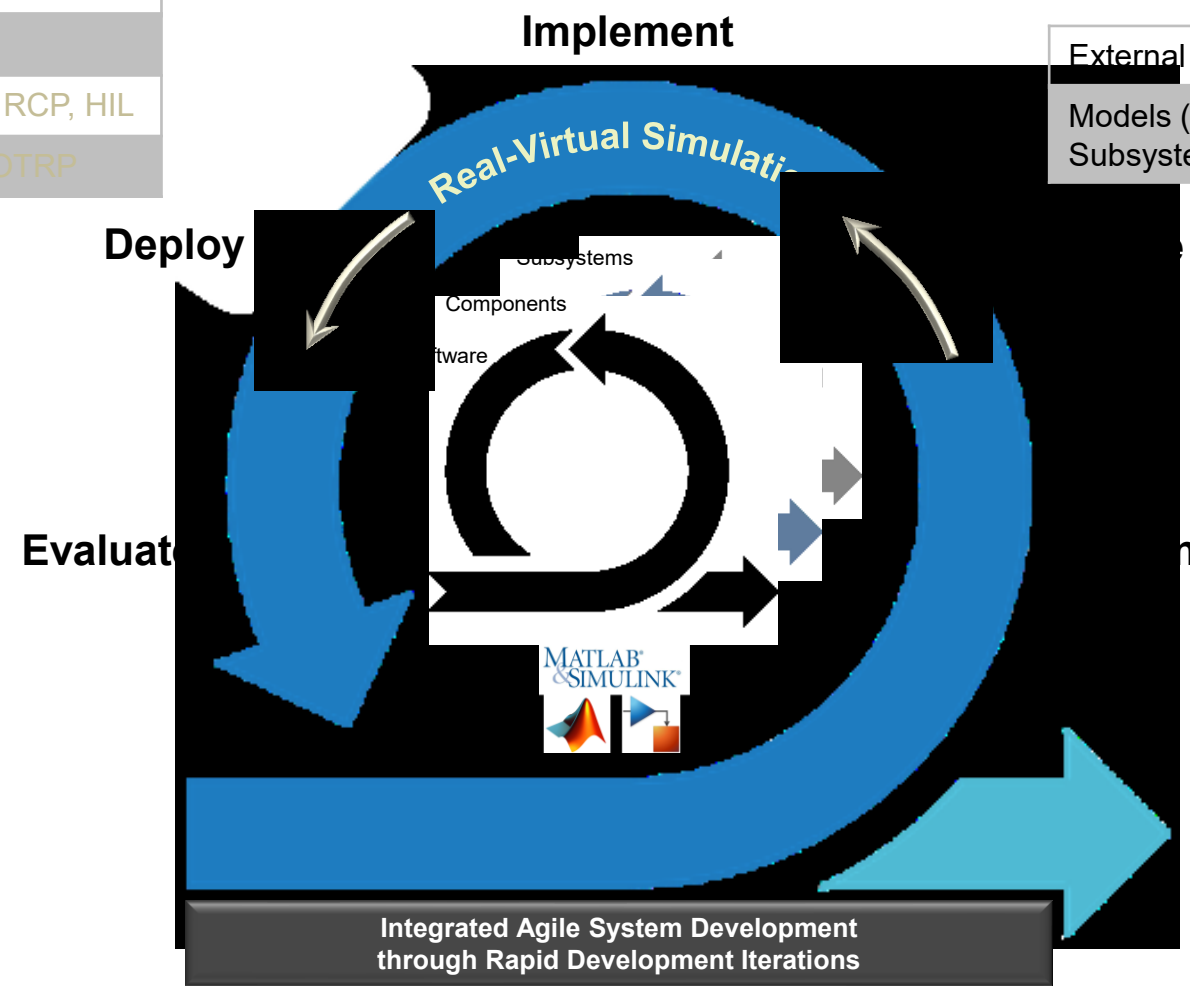


Model-Based Design

- Executable Specifications from Models
- Design with Simulation
- Implementation with Automatic Code Generation
- Continuous Test and Verification

Automatic Code Generation	Algorithm
	Plant

Desktop CPU: MIL, SIL
Target CPU: PIL
Prototyping RT CPU: RCP, HIL
Production EC: HIL, OTRP



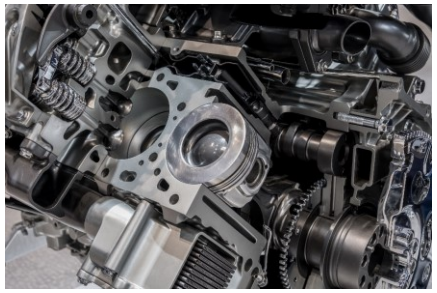
External Code
Models (Components, Subsystems, System)

	Controller	Plant
MIL, SIL, PIL	Virtual	Virtual
RCP	Virtual RT	Real
OTRP	Real	Real
HIL	Real	Virtual RT

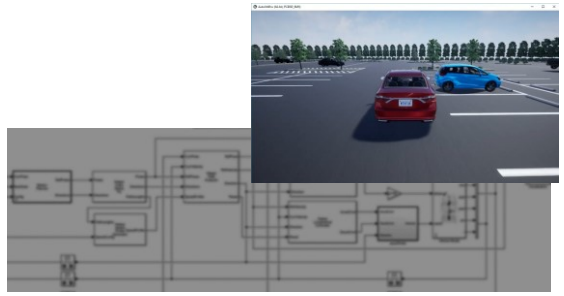
Algorithm
Plant Model
System Architecture

Who will be successful in the future?

Mechanical-centric



Model-centric



Software-centric

Comprehensive models
Simulation based testing
Generate code and automate verification

Questions?