Architecting Embedded Software Using Model-Based Design

Alan Moore

The MathWorks

Overview

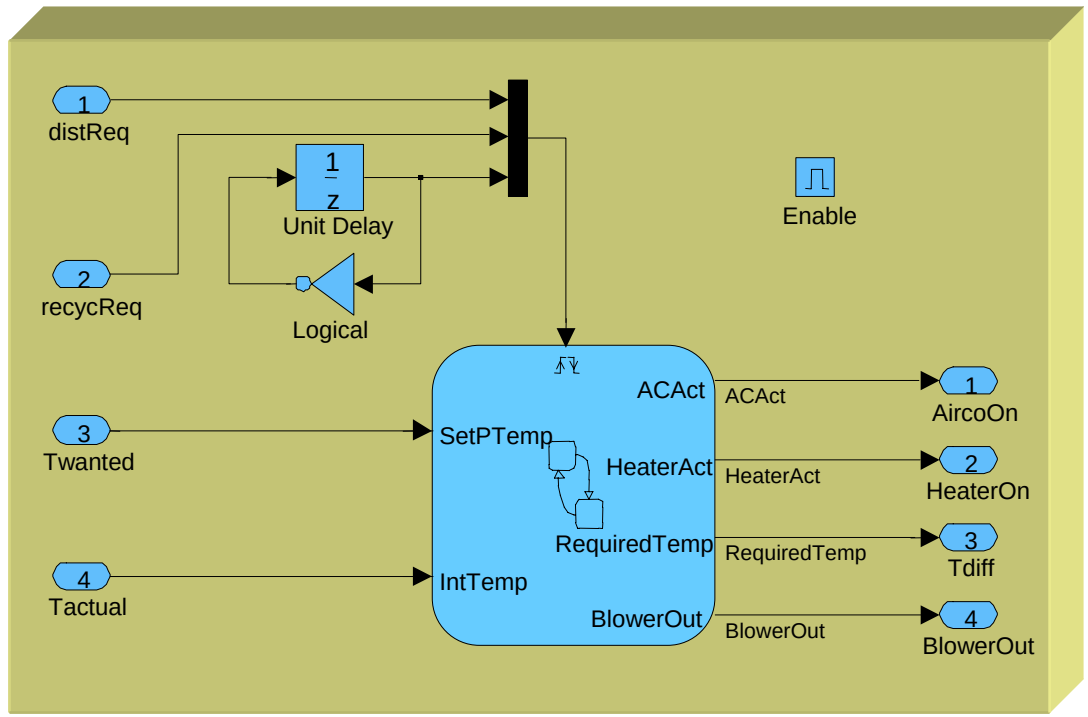
- Developing algorithmic models in Simulink
  - Composing algorithmic models within the Simulink algorithmic architecture
  - Deploying algorithmic models as software systems
- Integrating Simulink algorithmic models into a component-based architecture
  - Translating algorithmic models into software components in a software architecture
    - Programming language as integrating medium
    - UML as integrating medium
  - Integrating algorithmic models as components in an AUTOSAR software architecture

A Simple Algorithmic Model

- Twanted – desired temperature
- Tactual – sensed temperature
- recycReq – recycle air
- distReq – distribute heat

- ACAct – activate AC
- HeaterAct – activate heater
- RequiredTemp – heater temperature
- BlowerOut – fan speed

Temperature Control Algorithm



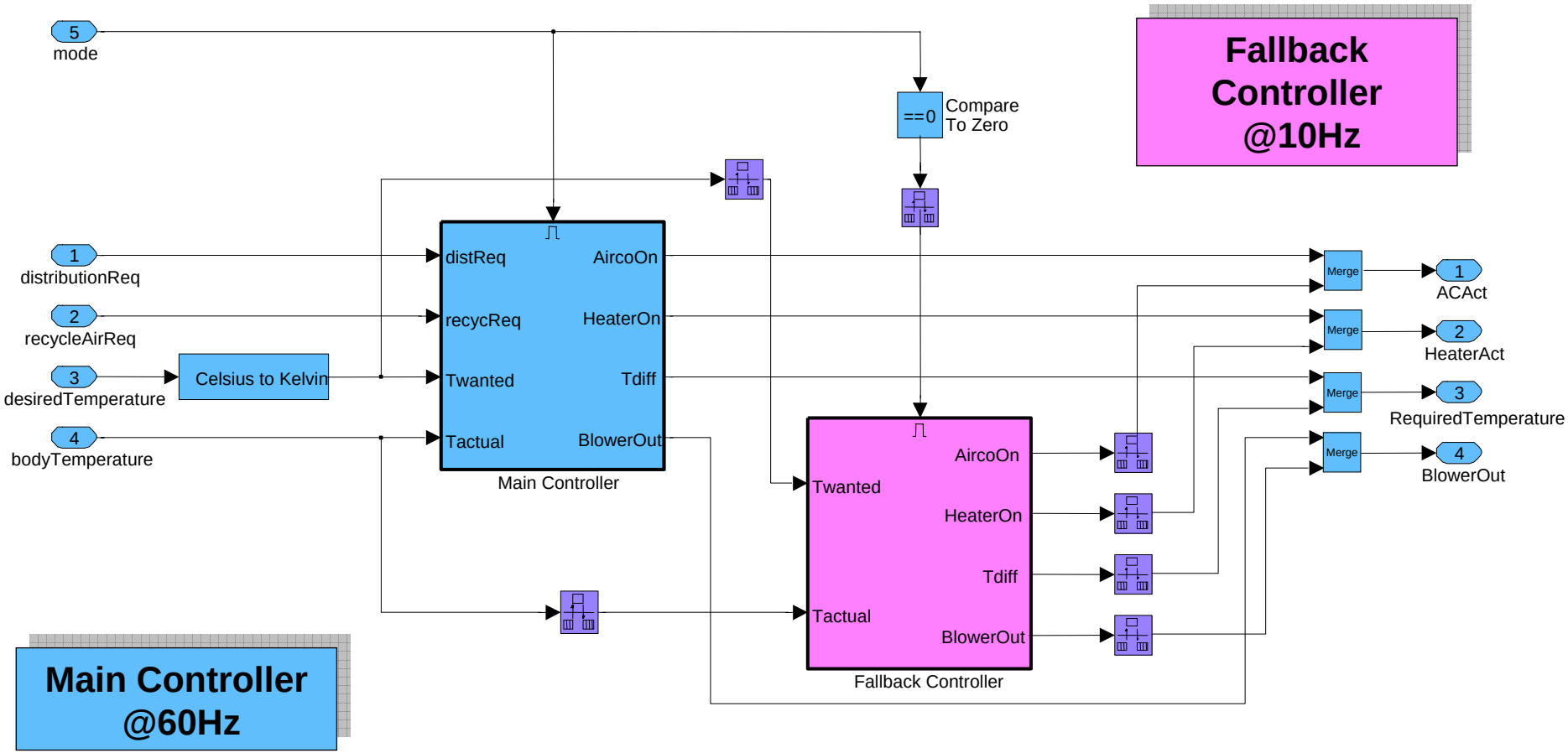
Technologies for Algorithm Composition

- **Subsystems**
  - Interface propagation supports iterative approach
  - Richer interface (enabled subsystem, function trigger ports)
  - For prototyping and smaller algorithms
- **Referenceable models**
  - Interface tightly specified so easier to export to other environments
  - More modular and scaleable
  - Standalone so easier to integrate into external CM projects
  - For larger algorithms and external deployment

- Both can be organized into libraries for reuse
- Automated conversion is available between the two representations

AM14 End with stuff about reuse.
also talk about smaller components/composites in choice section.
Alan Moore, 6/1/2007

Composite Algorithmic Model

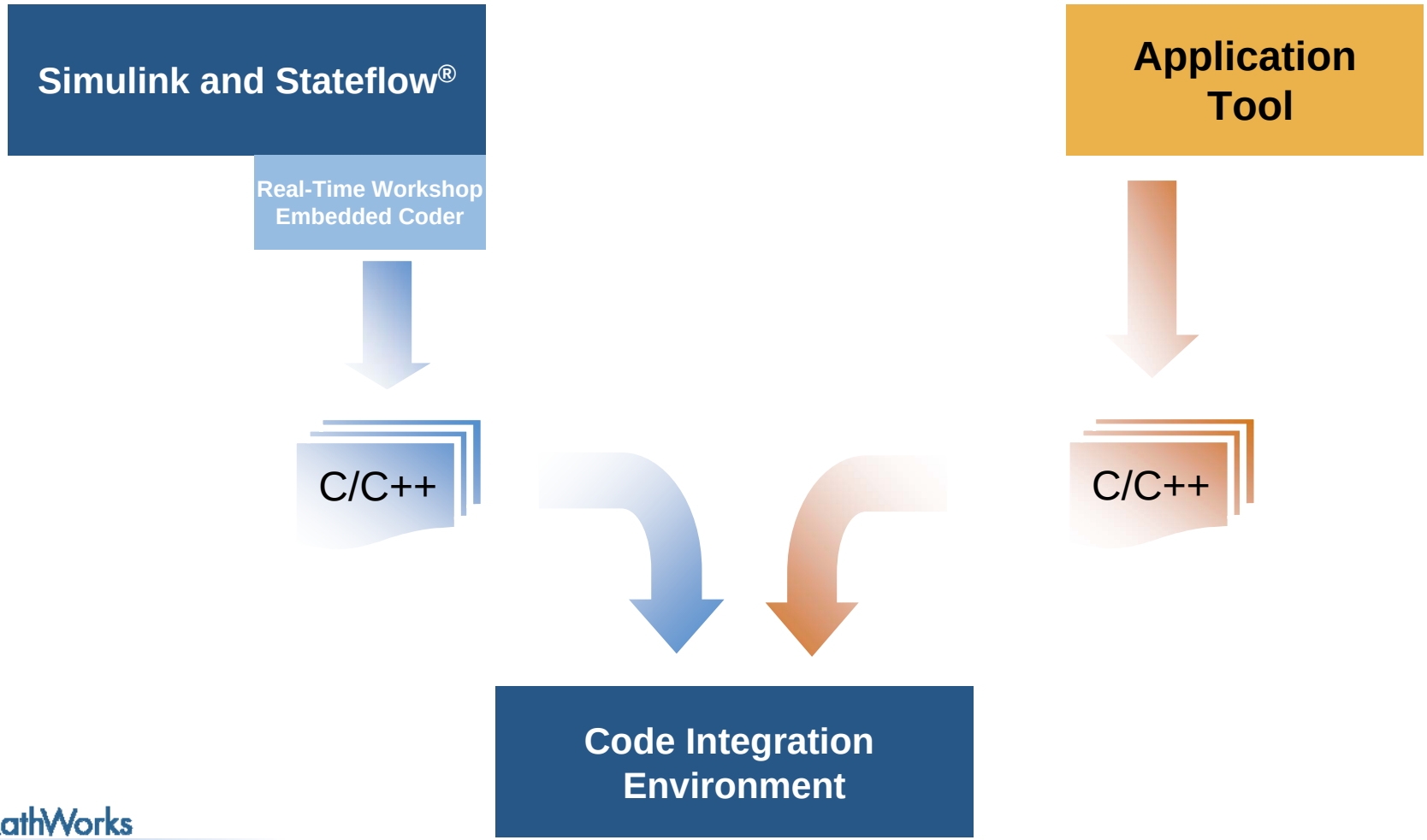


Route to Software

- More simple software architectures
  - Rate-monotonic schedule
  - Flow-based architecture
  - Generate software system using Real-Time Workshop Embedded Coder
- More complex software architectures
  - Service-based architecture
  - Mandatory architectural patterns
    - Error handling
    - Supervision
    - …
  - Generate software component using Real-Time Workshop Embedded Coder

More Complex Software Architectures

Architectural Integration Through Code



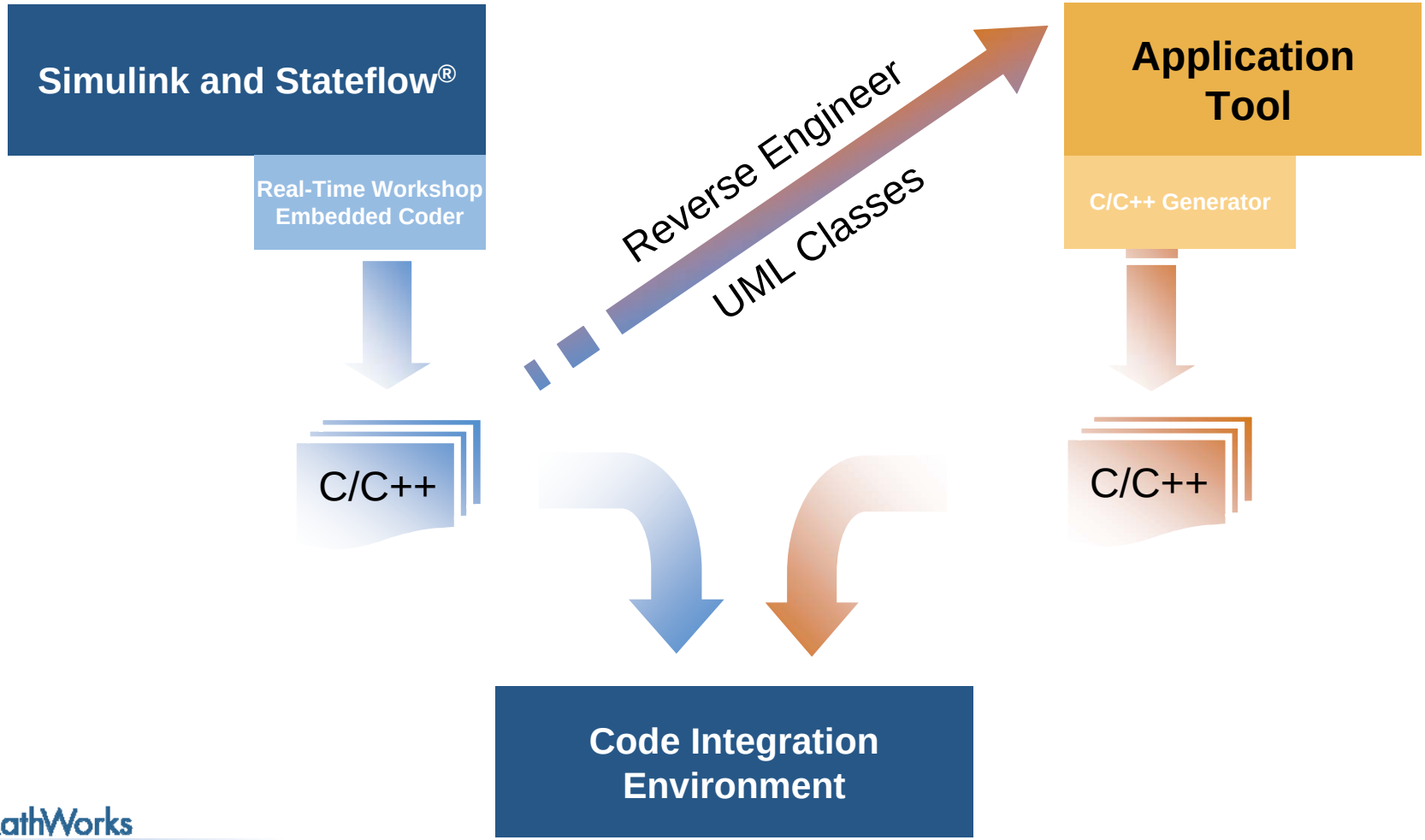
More Complex Software Architectures

The Unified Modeling Language (UML)

- Maintained by the Object Management Group (OMG)
  - Version 1.0 published in 1997
  - Version 2.0 published in 2005
  - Currently at Version 2.1.1
- Highlights
  - Wide variety of diagrams to support many phases of software development, including architecture and deployment
  - UML 1.X oriented around class (object-oriented) modeling
  - UML 2.X introduced more component-based modeling concepts
  - Extensions to UML under development for real-time and embedded systems

More Complex Software Architectures

UML Integration Through Code



AUTOSAR Overview

- AUTOSAR Goals
  - Implementation and standardization of a single platform as an OEM wide “Standard Core” solution
  - Enable OEM’s to focus on added value
- AUTOSAR Status
  - AUTOSAR Release 2.0 specifications
    - Published in May 2006, for information only
    - Available for download – www.autosar.org
  - AUTOSAR Release 2.1
    - Scheduled for end of 2006
    - Will also be published and available for download

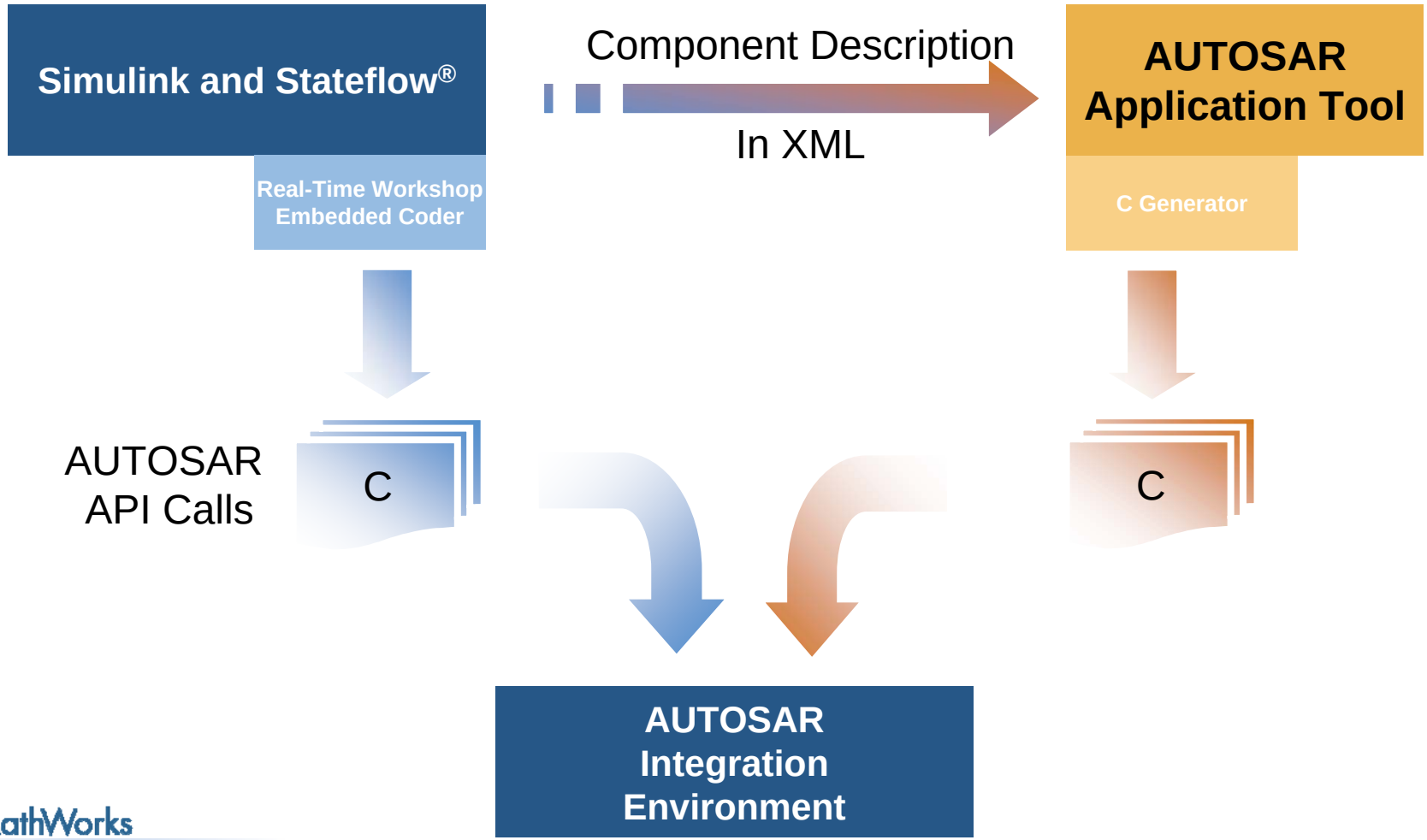
AUTOSAR Key Technologies*

- Basic Software
  - Software architecture including a complete basic (environmental) software stack for an ECU as an integration platform for hardware independent SW applications
- Methods of Software Integration
  - Exchange formats (templates) to enable a seamless configuration process of the basic software stack and the integration of application software in ECUs
- Functional API
  - Specification of functional interfaces as a standard for application software modules

*Source: Helmut Fennel, OOP 2007

More Complex Software Architectures

AUTOSAR Integration



More Complex Software Architectures

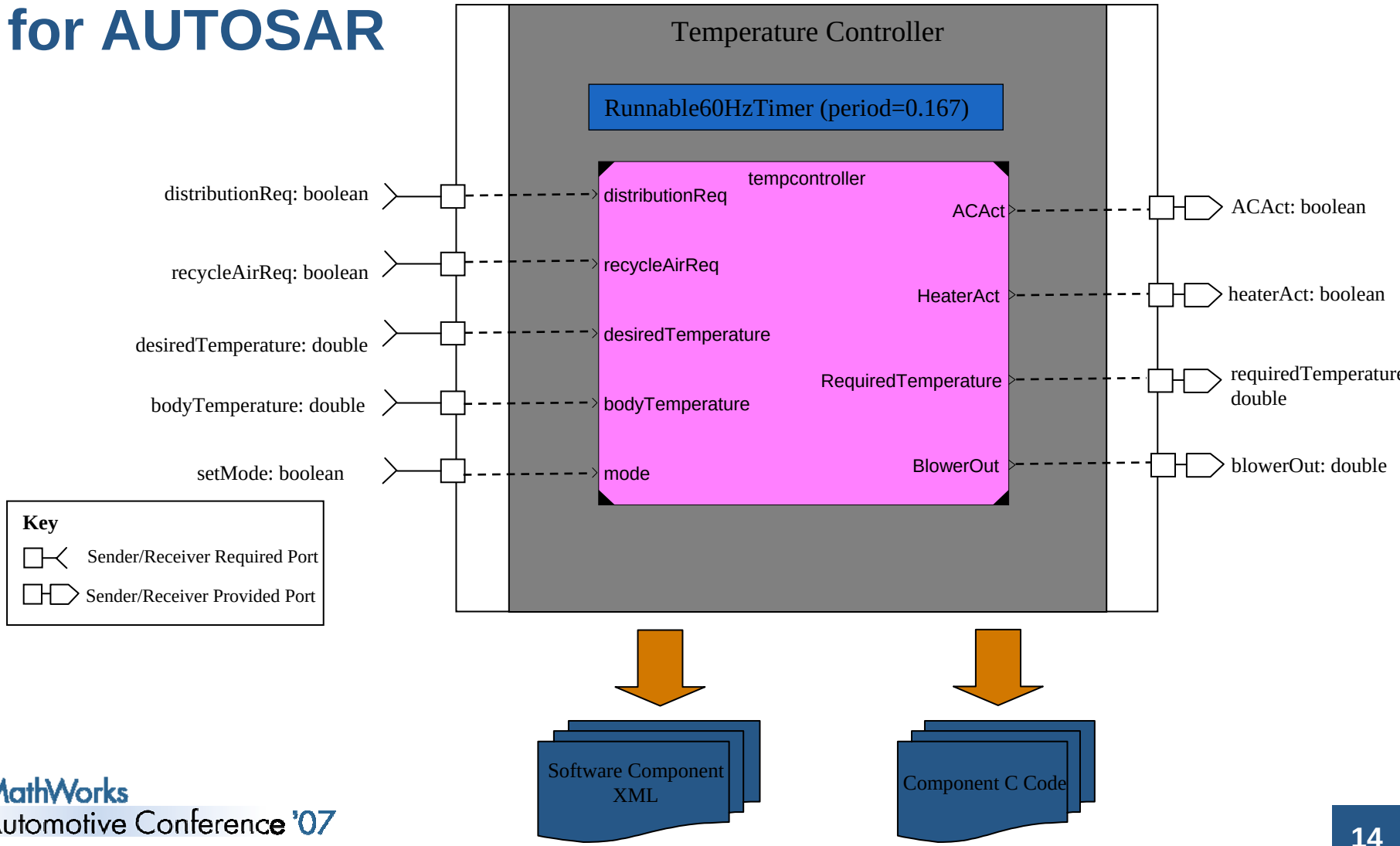
AUTOSAR Demonstration Kit (ADK)

- Uses Simulink to import and export:
  - AUTOSAR Software Component (SW-C) Descriptions, in XML
- Customizes Real-Time Workshop® Embedded Coder to generate:
  - AUTOSAR SW-C Implementations (runnables) compliant with AUTOSAR Run Time Environment, in C code
- Supports:
  - AUTOSAR v2.0 and v2.1
  - Simulink R2006b and R2007a

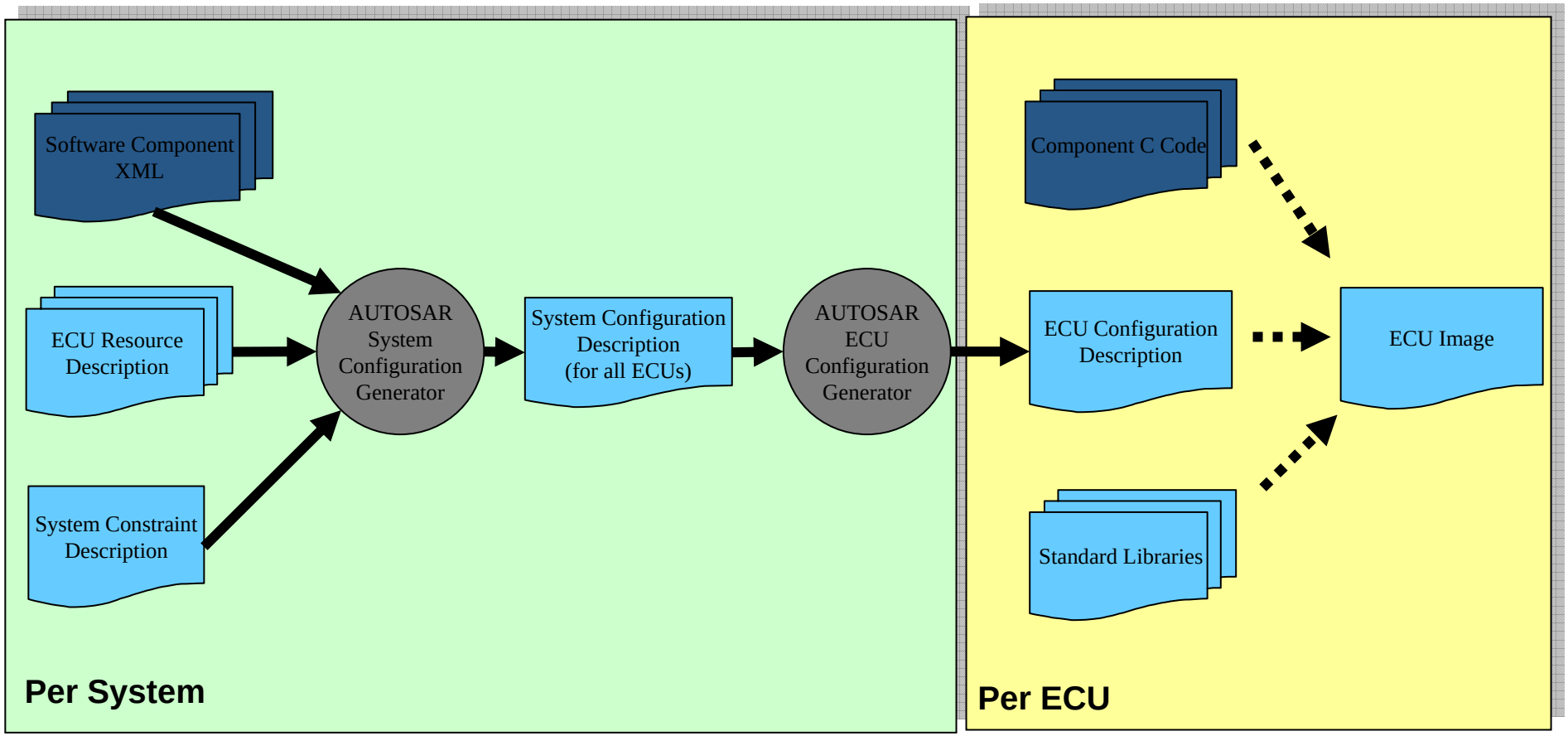
Is a work in progress so is subject to change

More Complex Software Architectures

Temperature Control Algorithm Wrapped for AUTOSAR



Use of ADK Artifacts in AUTOSAR



N.B. Only part of the total AUTOSAR tool chain shown here

Conclusion

- Simulink is the established architectural environment for algorithmic development
  - Choice of architectural approaches
  - Rich design and verification environment
  - Route to production code
- Software architectures are becoming more complex
  - Need to publish algorithmic models as components for integration
  - Real-Time Workshop Embedded Coder offers flexible C/C++ generation to create software components for integration
  - A C/C++ code-based approach can also be taken where UML is used for the software architecture
- Domain-specific architectures need more specialist support
  - AUTOSAR is maturing as a component-based platform in the automotive domain
  - AUTOSAR requires additional artifacts besides code to drive the AUTOSAR tool chain
  - The AUTOSAR Demonstration Kit allows a Simulink algorithmic model to be published as an AUTOSAR software component