

Model-based Design and Implementation of Distributed Real-time Embedded Systems in Cadena

[*http://www.cis.ksu.edu/cadena*](http://www.cis.ksu.edu/cadena)

Gurdip Singh

John Hatcliff

Matt Dwyer

Venkatesh Ranganath

Xianghua Deng

Prashant Shanti Kumar

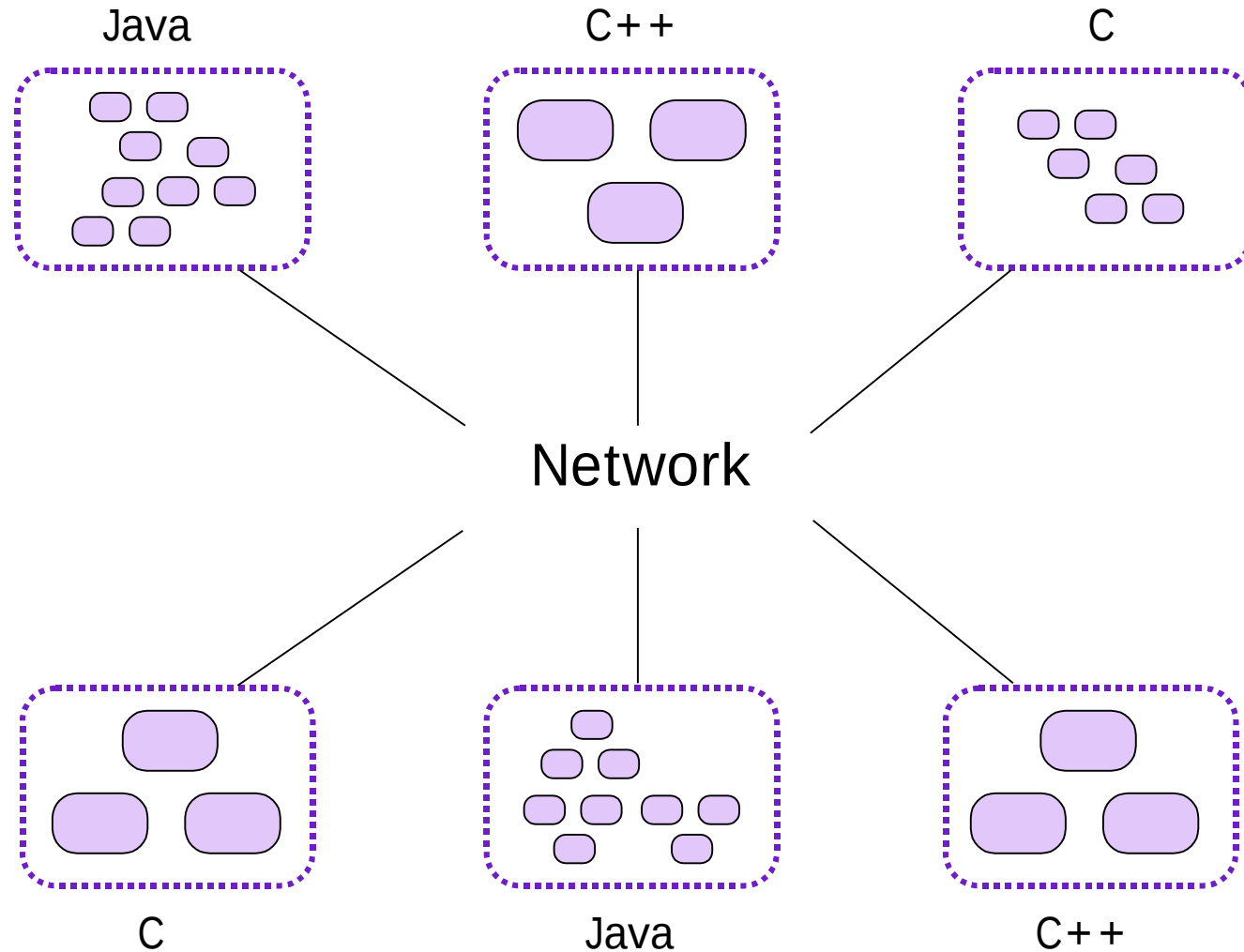
Qiang Deng

Support

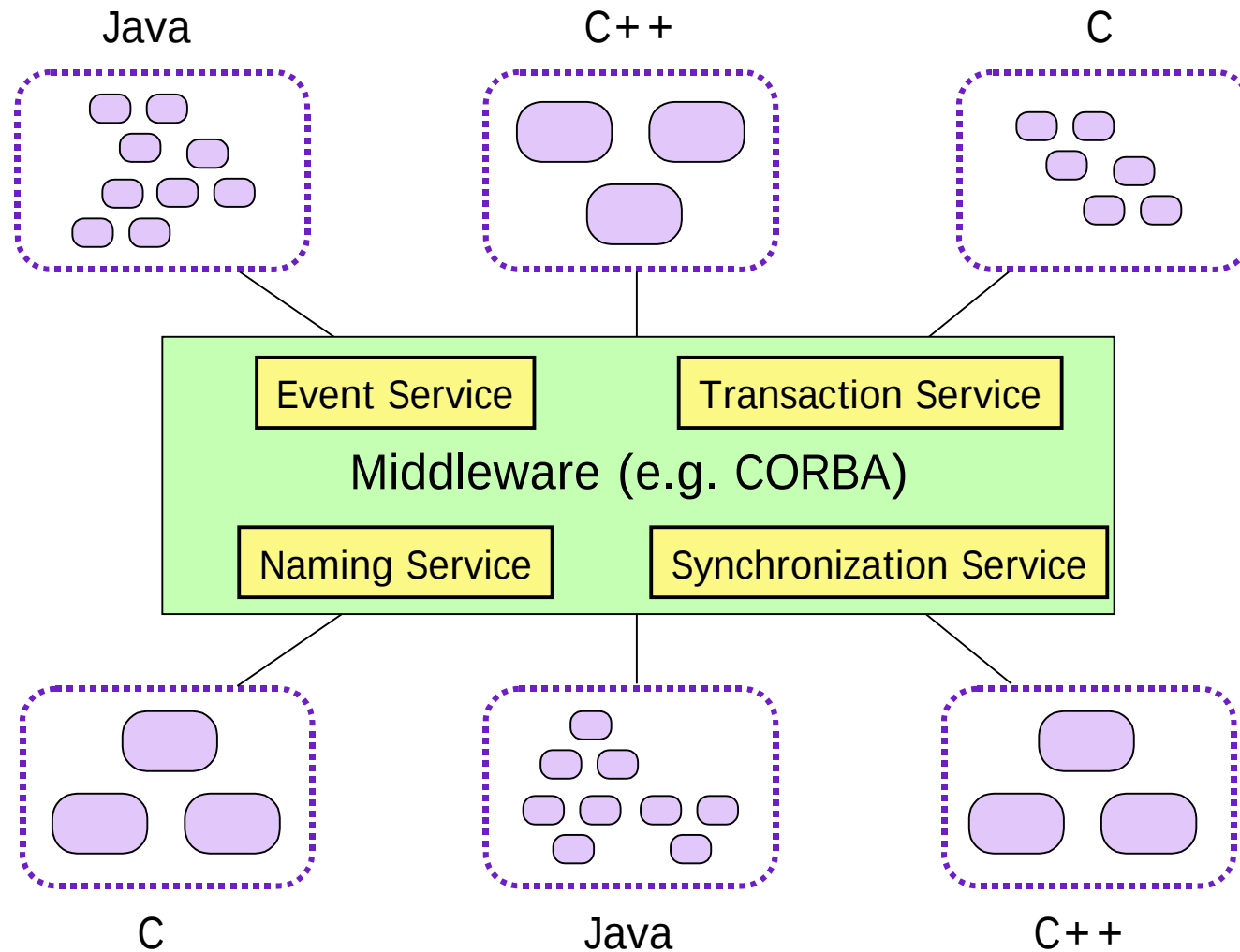
US Army Research Office (ARO)
US National Science Foundation (NSF)
US National Aeronautics and Space Agency (NASA)
US Department of Defense
Advanced Research Projects Agency (DARPA)

Rockwell-Collins ATC
Honeywell Technology Center and NASA Langley
Sun Microsystems
Intel

Distributed Components

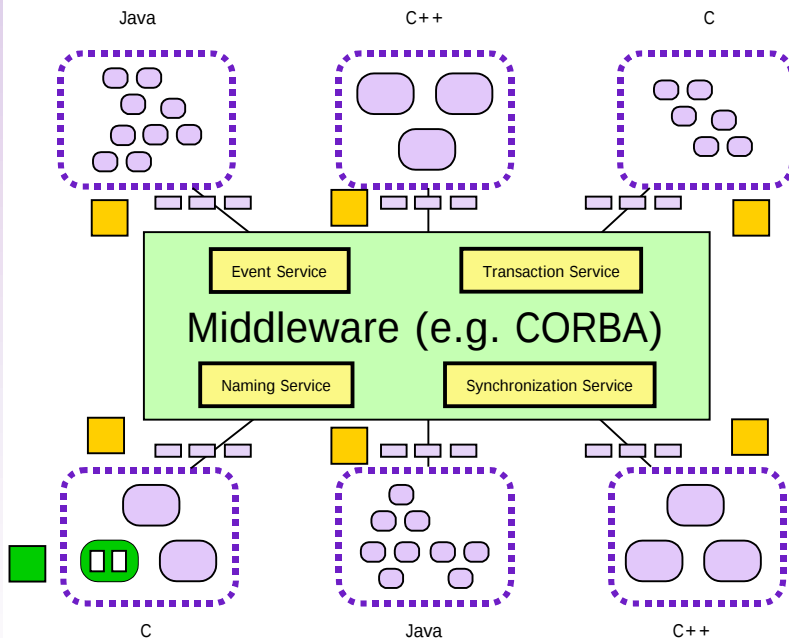


Distributed Components



Real-time Embedded Systems

Issues



- These systems are huge!
- Extensive use of OO patterns & software layering
- What are appropriate abstractions for formal reasoning?
- How can we help developers write them?
- How to auto-generate aspects?
- How can we configure middleware?

Example DRE Systems



- Mission-control software for Boeing military aircraft
- Boeing's Bold Stroke Avionics Middleware
 - ...built on top of ACE/TAO RT CORBA

- Provided with an Open Experimental Platform (OEP) from Boeing
 - a sanitized version of the real system
 - 100,000+ lines of C++ code (including RT CORBA middleware)
 - 50+ page document that outline development process and describe challenge problems
- Must provide...
 - tool-based solutions that can be applied by Boeing research team to realistic systems
 - solutions that fit within current development process, code base, etc.
 - metrics for that allow Boeing research team to evaluate tool performance and ease of use

Cadena – CCM Development

Cadena

CCM Interface
Definition
Language

RT Aspect Specs
State Transitions

System
Configuration

High-level
Specification Language

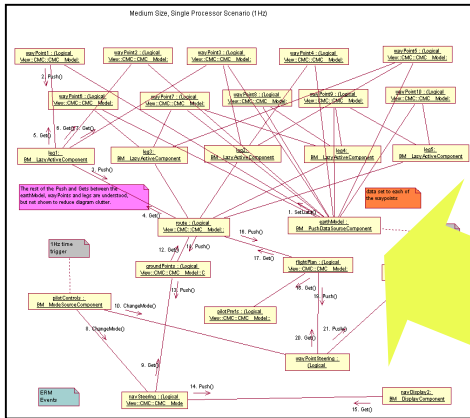
Eclipse Plug-In

Integrated Development
Environment

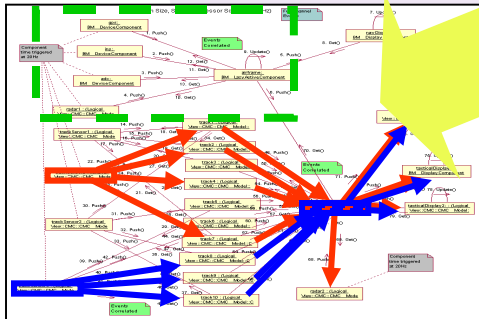
Code generation functions
(via OpenCCM) produces
code amenable to
conformance checking and
certification

```
...URATION_PASS>
...<...>
...COMPONENT>
...ID> <...></ID>
<EVENT_SUPPLIER>
<...events this component supplies...>
</EVENT_SUPPLIER>
</COMPONENT>
</HOME>
</CONFIGURATION_PASS>
```

Configuration and
Deployment information



High-level specification of
abstract component
behavior



Visualization and
design-level reasoning

Overall Themes

DRE Systems
in a CCM
Framework

Component
Development

Scenario
Specification

Scenario
Analysis

Scenario
Deployment

Middleware
Library



Themes

- ...COTS/Industry standard component models/frameworks
- ...Component developer's activities (interface & behavior specs) are centered at the CCM IDL level.

Overall Themes

DRE Systems
in a CCM
Framework

Component
Development

Scenario
Specification

Scenario
Analysis

Scenario
Deployment

Middleware
Library



Themes

...Specify and reason about component connections and dependencies independently of middleware services

Overall Themes

DRE Systems
in a CCM
Framework

Component
Development

Scenario
Specification

Scenario
Analysis

Scenario
Deployment

Middleware
Library



Themes

- ...Choose particular middleware services and synthesize QoS attributes.
- ...Check invariants, dependencies, and temporal properties.

Overall Themes

DRE Systems
in a CCM
Framework

Component
Development

Scenario
Specification

Scenario
Analysis

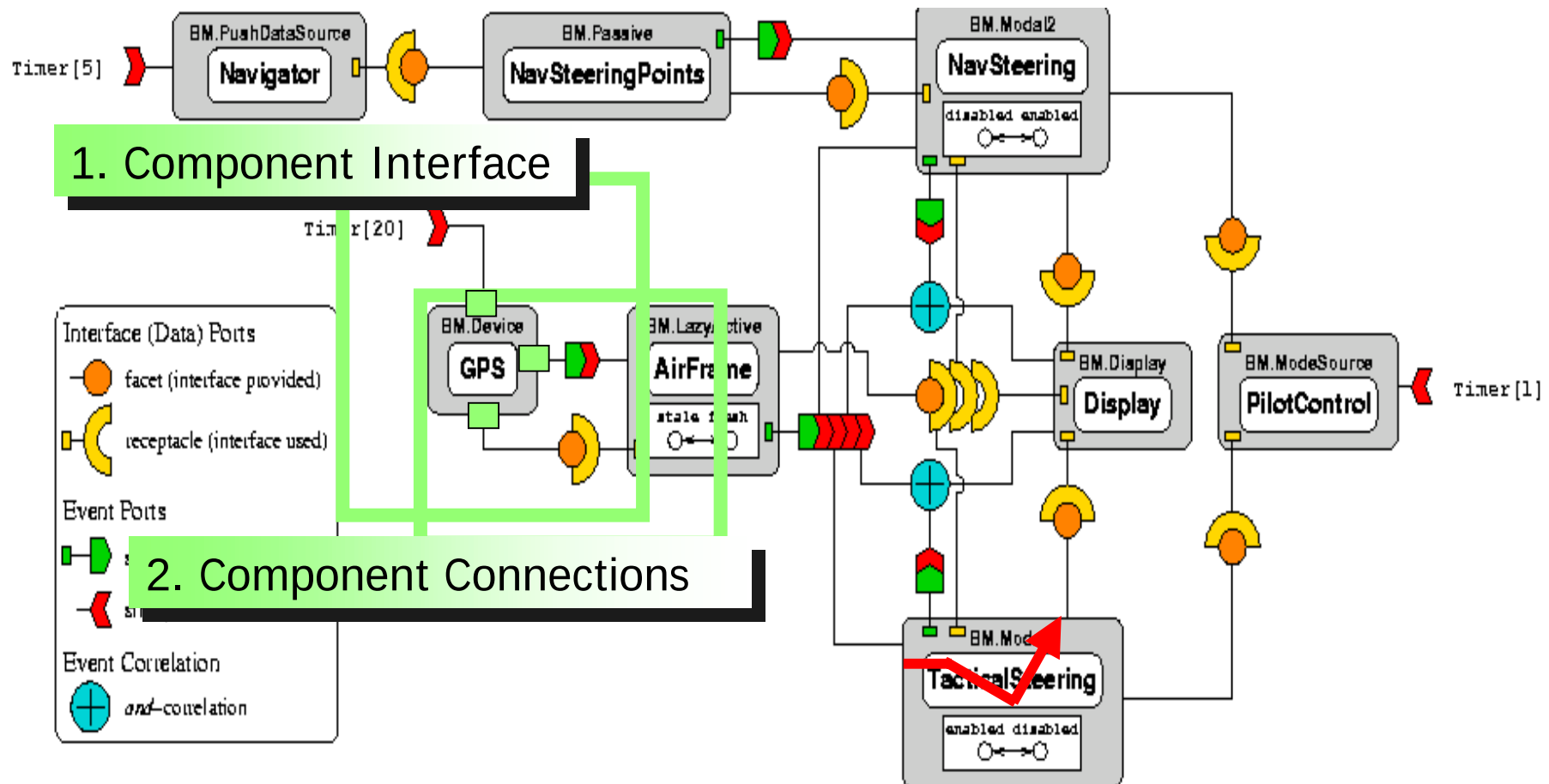
Scenario
Deployment

Middleware
Library

Themes

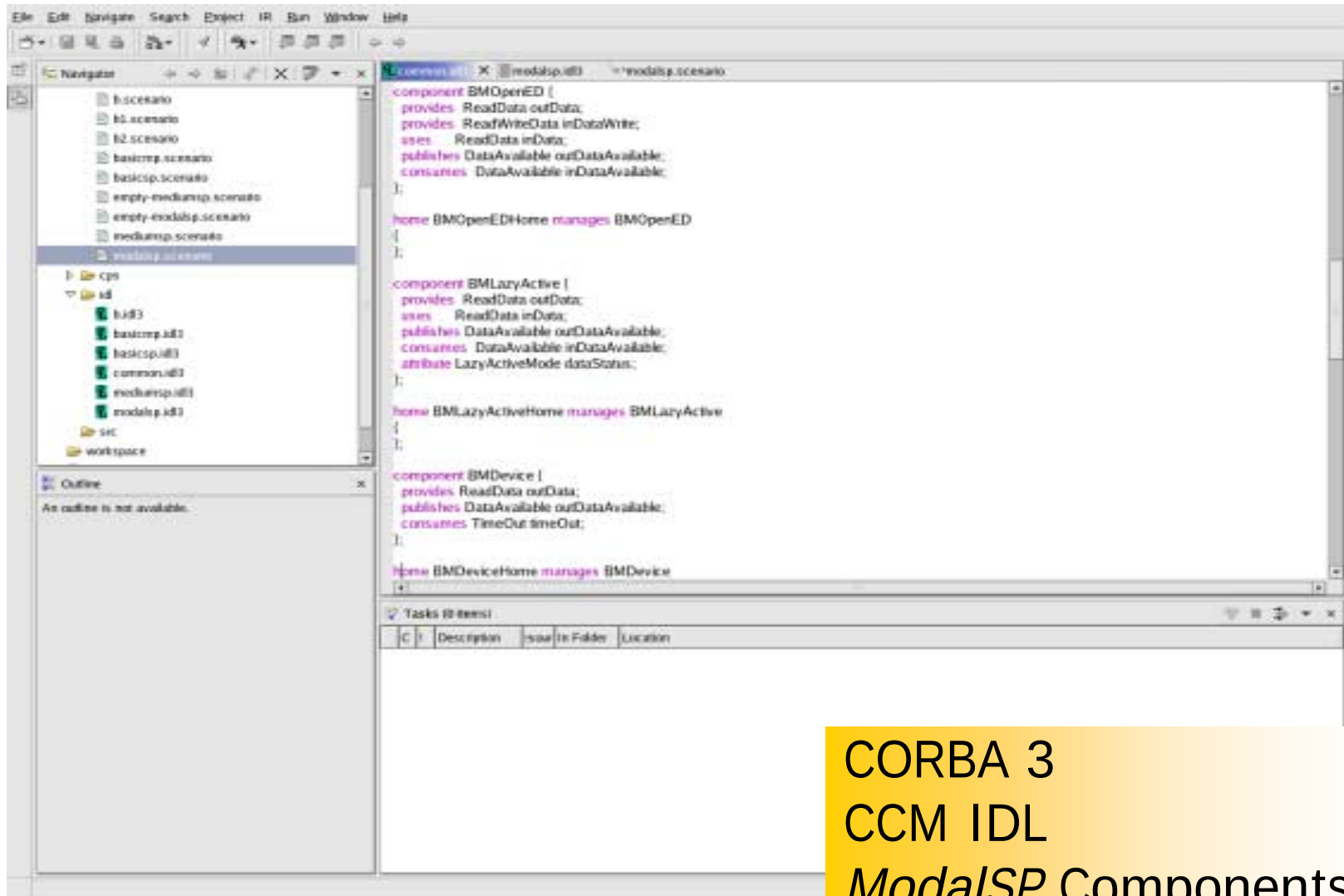
- ...Autogenerate configuration and component container code for chosen middleware services.
- ...Deploy based on deployment parameters.

Outline



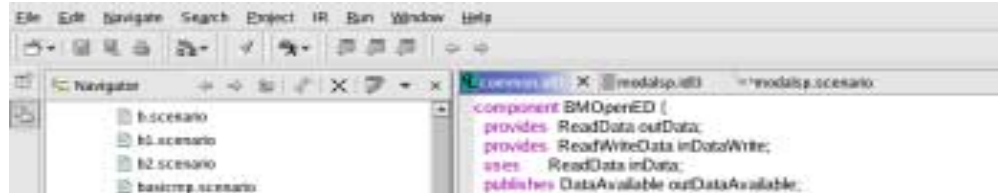
3. Dependence Information

Component IDL

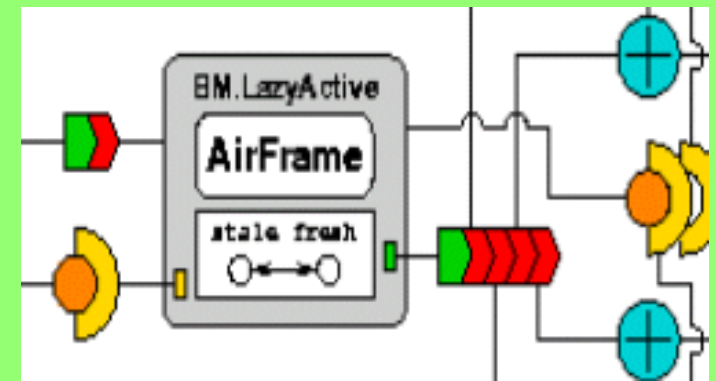


CORBA 3
CCM IDL
ModalSP Components

Component IDL

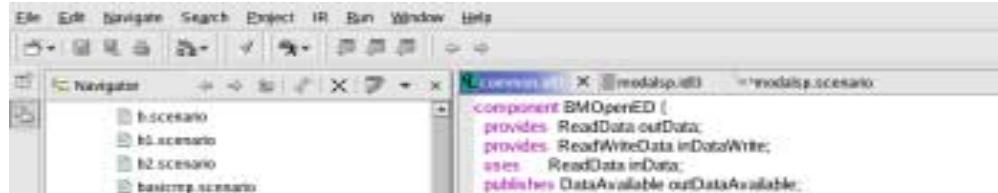


```
component BMLazyActive {  
  provides ReadData outData;  
  uses ReadData inData;  
  publishes DataAvailable outDataAvailable;  
  consumes DataAvailable inDataAvailable;  
  attribute LazyActiveMode dataStatus;  
};
```

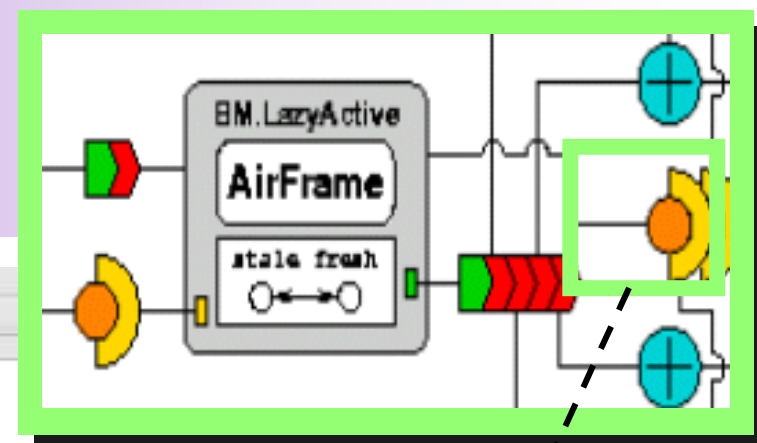


CORBA 3
CCM IDL
ModalSP Components

Component IDL



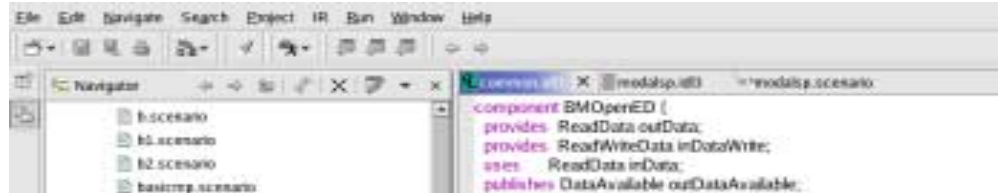
```
component BMLazyActive {  
  provides ReadData outData;  
  uses ReadData inData;  
  publishes DataAvailable outDataAvailable;  
  consumes DataAvailable inDataAvailable;  
  attribute LazyActiveMode dataStatus;  
};
```



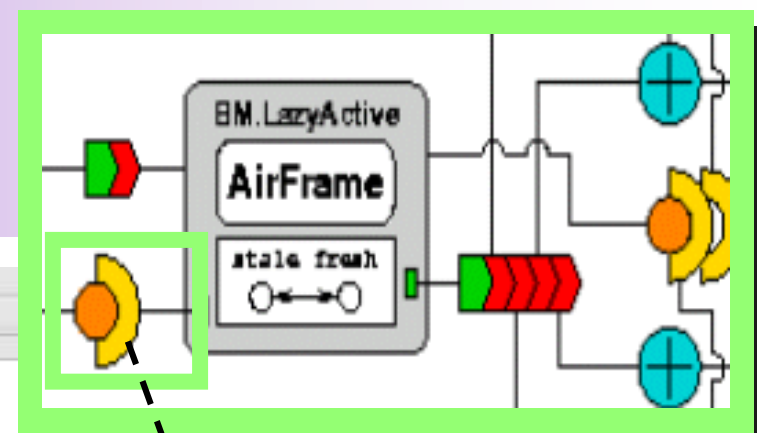
*output data port
(facet)*

CORBA 3
CCM IDL
ModalSP Components

Component IDL



```
component BMLazyActive {  
  provides ReadData outData;  
  uses ReadData inData;  
  publishes DataAvailable outDataAvailable;  
  consumes DataAvailable inDataAvailable;  
  attribute LazyActiveMode dataStatus;  
};
```

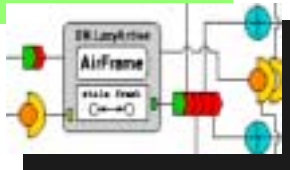


*input data port
(receptacle)*

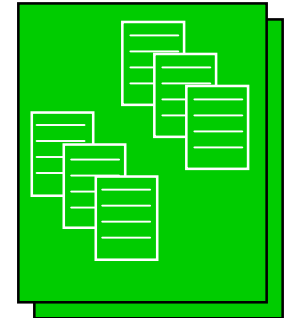
CORBA 3
CCM IDL
ModalSP Components

Leverage CO R B A IDL

```
component BMLazyActive {  
  provides ReadData outData;  
  uses ReadData inData;  
  publishes DataAvailable outDataAvailable;  
  consumes DataAvailable inDataAvailable;  
  attribute LazyActiveMode dataStatus;  
};
```



IDL Compiler



Component
Implementation
Stubs & Skeletons

+

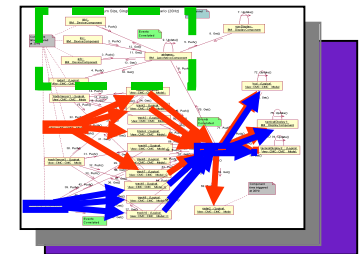
```
dependencydefault  
  == none;  
dependencies {  
  inDataAvailable  
  ->  
  outDataAvailable;  
}
```

Dependency
Annotations

```
behavior {  
  if (mode==enabled) {  
    push outDataAvailable;  
  }  
  ...  
}
```

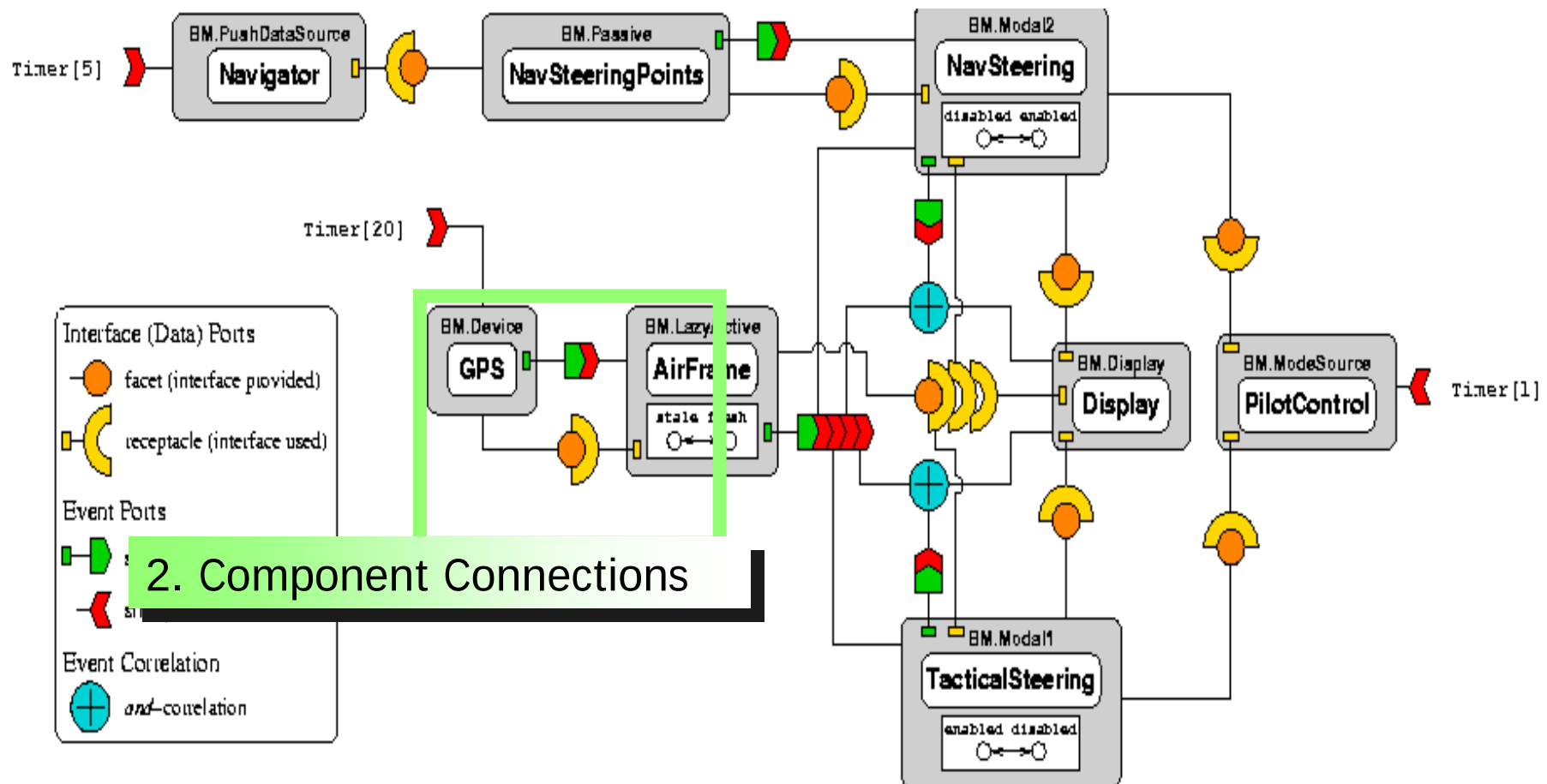
Transition System
Semantics

Model Builder



Dependency Analysis
and
Model-checking Engine

Outline



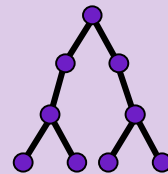
Three Synchronized Views

Scenario
Description

Graphical
View

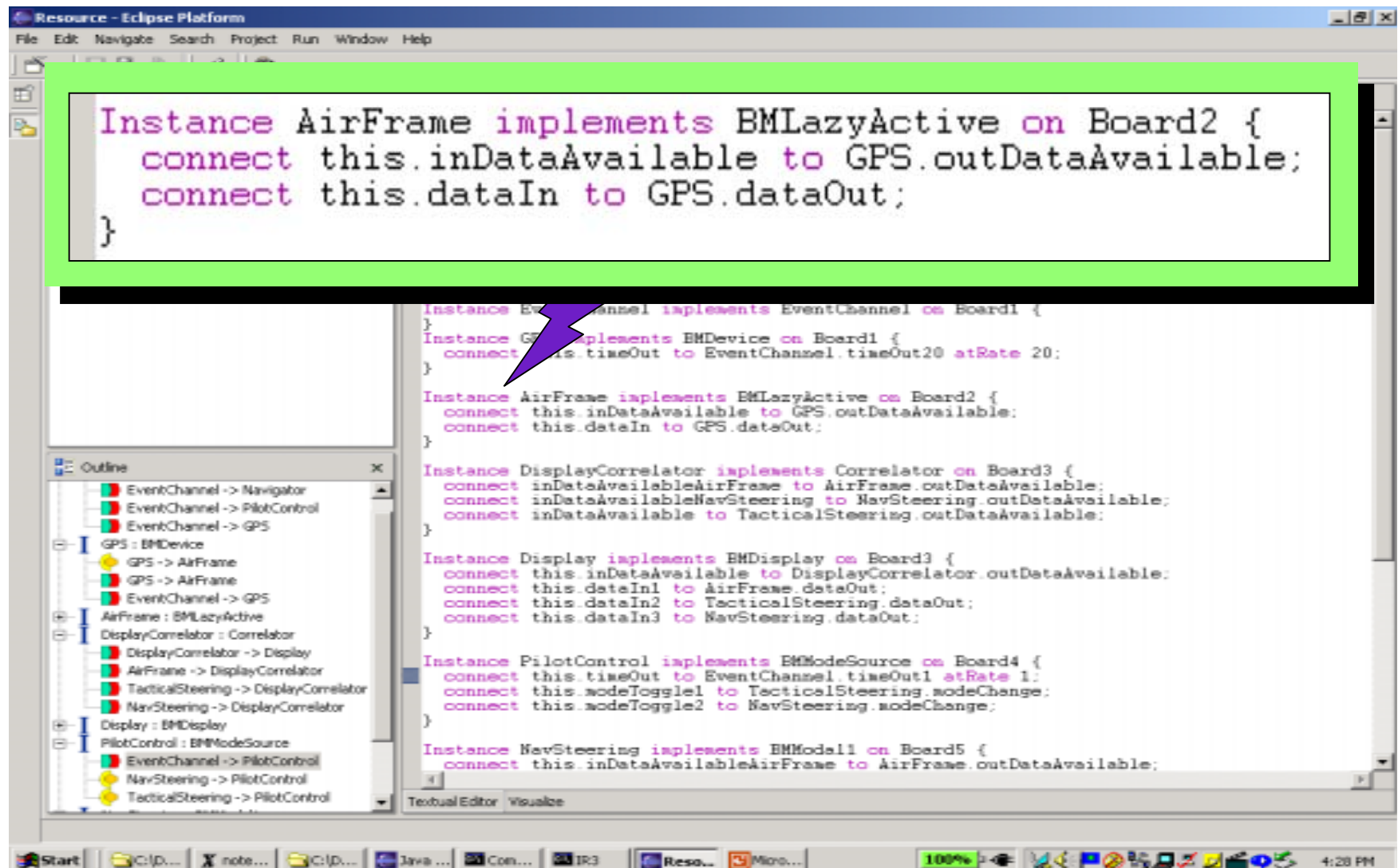
Spreadsheet
View

Textual
View



Single
Internal
Representation

Textual View



Resource - Eclipse Platform

File Edit Navigate Search Project Run Window Help

```
Instance AirFrame implements BMLazyActive on Board2 {  
    connect this.inDataAvailable to GPS.outDataAvailable;  
    connect this.dataIn to GPS.dataOut;  
}
```

Instance EventChannel implements EventChannel on Board1 {
}
Instance GPS implements BMDevice on Board1 {
 connect this.timeOut to EventChannel.timeOut20 atRate 20;
}
Instance AirFrame implements BMLazyActive on Board2 {
 connect this.inDataAvailable to GPS.outDataAvailable;
 connect this.dataIn to GPS.dataOut;
}
Instance DisplayCorrelator implements Correlator on Board3 {
 connect inDataAvailableAirFrame to AirFrame.outDataAvailable;
 connect inDataAvailableNavSteering to NavSteering.outDataAvailable;
 connect inDataAvailable to TacticalSteering.outDataAvailable;
}
Instance Display implements BMDisplay on Board3 {
 connect this.inDataAvailable to DisplayCorrelator.outDataAvailable;
 connect this.dataIn1 to AirFrame.dataOut;
 connect this.dataIn2 to TacticalSteering.dataOut;
 connect this.dataIn3 to NavSteering.dataOut;
}
Instance PilotControl implements BMModeSource on Board4 {
 connect this.timeOut to EventChannel.timeOut1 atRate 1;
 connect this.modeToggle1 to TacticalSteering.modeChange;
 connect this.modeToggle2 to NavSteering.modeChange;
}
Instance NavSteering implements BMModel1 on Board5 {
 connect this.inDataAvailableAirFrame to AirFrame.outDataAvailable;

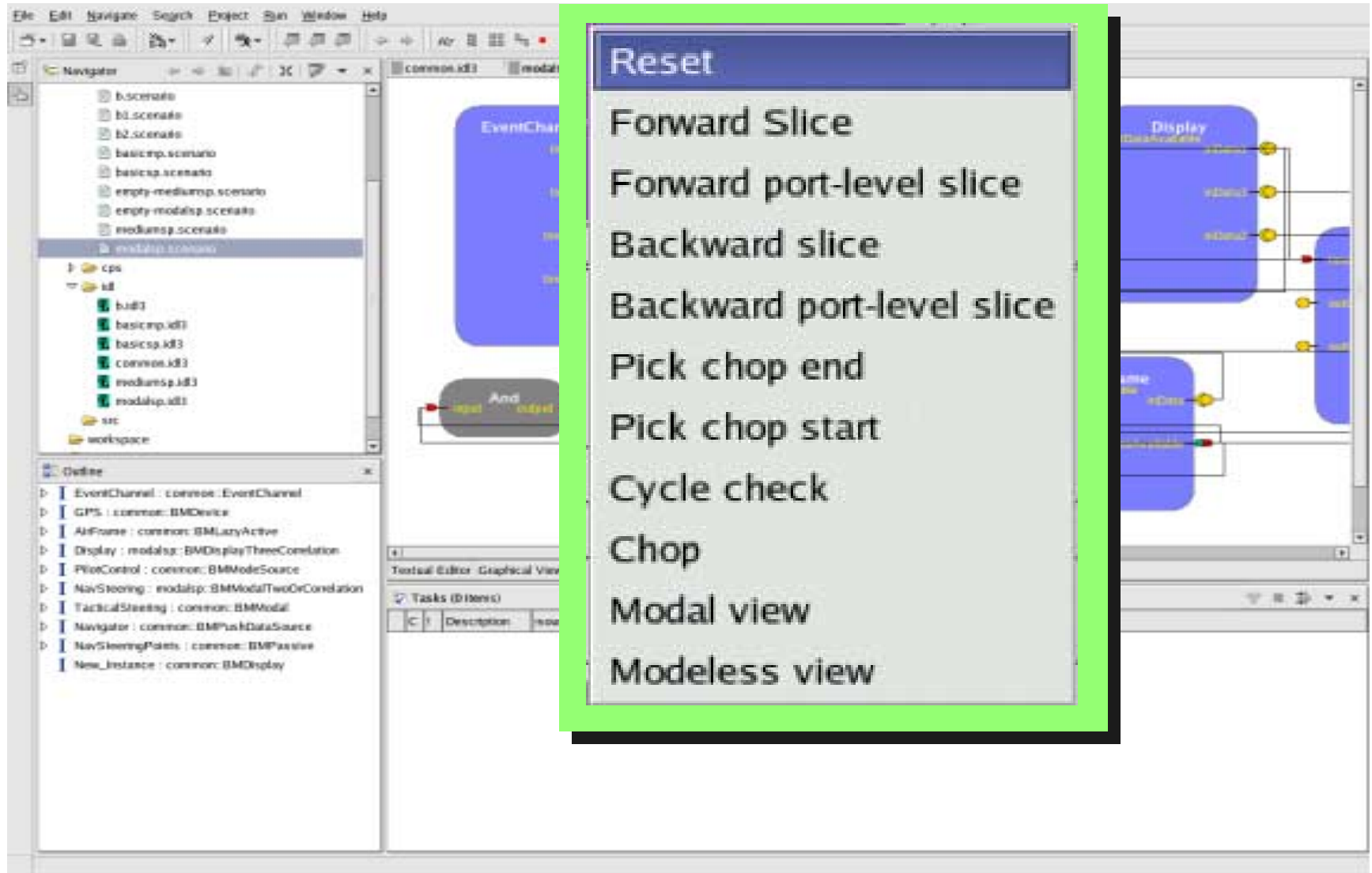
Outline

- EventChannel -> Navigator
- EventChannel -> PilotControl
- EventChannel -> GPS
- GPS : BMDevice
 - GPS -> AirFrame
 - GPS -> AirFrame
- EventChannel -> GPS
- AirFrame : BMLazyActive
- DisplayCorrelator : Correlator
 - DisplayCorrelator -> Display
 - AirFrame -> DisplayCorrelator
 - TacticalSteering -> DisplayCorrelator
 - NavSteering -> DisplayCorrelator
- Display : BMDisplay
- PilotControl : BMModeSource
 - EventChannel -> PilotControl
 - NavSteering -> PilotControl
 - TacticalSteering -> PilotControl

Textual Editor Visualize

Start C:\D... note... C:\D... Java... Con... TR3 Reso... Micro... 100% 4:28 PM

Graphical View



Spreadsheet View



GPS	Board1				
timeOut <	::models::TimeOut	20	< timeOut20	EventChannel	
outDataAvailable >	::models::DataAvailable	20	> inDataAvailable	AirFrame	
dataOut >	::models::ReadData	0	> dataIn	AirFrame	

...ports for
component
type

...port types

RT Attributes

...port connections

...distribution
sites

...rate
group

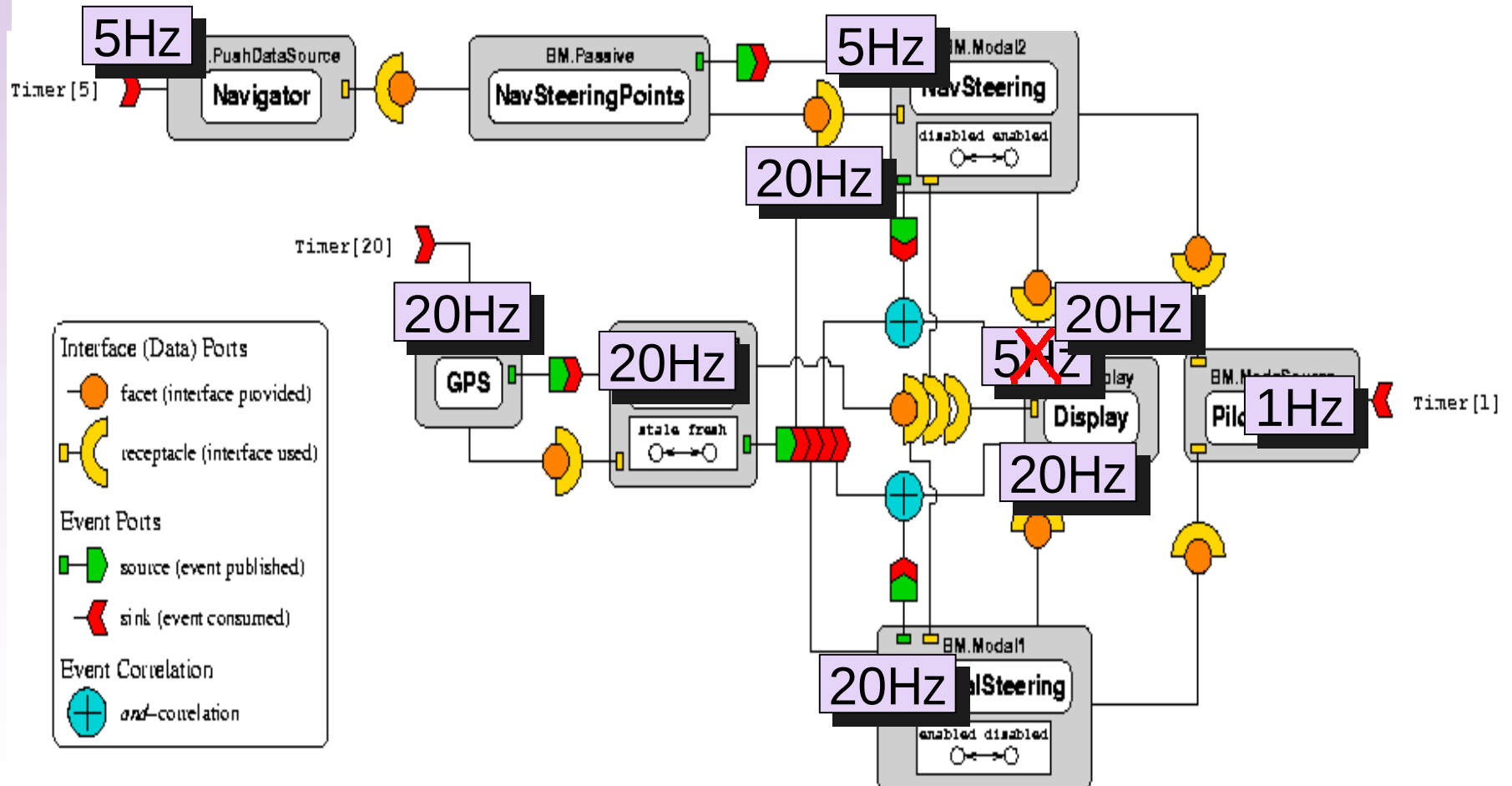
EventChannel : models::EventChannel					
GPS : models::BMDDevice					
AirFrame : models::BPLazyActive					
DisplayCorrelator : models::Correlator					
Display : models::BMDisplay					
PilotControl : models::BMDModeSource					
NavSteering : models::BMDMode2					
TacticalSteering : models::BMDModel1					
Navigator : models::BMDPushDataSource					
NavSteeringPoints : models::BMDPassive					
inDataAvailableAirFrame <		20	< outDataAvailable	AirFrame	
inDataAvailable <		20	< outDataAvailable	TacticalSteering	
dataIn2 <		20	< dataOut1	TacticalSteering	
dataIn3 <		20	< dataOut	NavSteering	
inDataAvailable <		20	< outDataAvailable	DisplayCorrelator	
dataIn1 <		20	< dataOut	AirFrame	
timeOut <	::models::TimeOut	1	< timeOut1	EventChannel	
modeToggle2 <	::models::ChangeMode	1	< modeChange	NavSteering	
modeToggle1 <	::models::ChangeMode	1	< modeChange	TacticalSteering	
dataIn2 <	::models::ReadData	20	< dataOut	AirFrame	
inDataAvailableTacticalSteering <	::models::DataAvailable	0	< outDataAvailable	NavSteeringPoints	
outDataAvailable >	::models::DataAvailable	20	> inDataAvailableNavSteer...	DisplayCorrelator	
inDataAvailableAirFrame <	::models::DataAvailable	20	< outDataAvailable	AirFrame	
modeChange >	::models::ChangeMode	0	> modeToggle2	PilotControl	
dataIn1 <	::models::ReadData	20	< dataOut	NavSteeringPoints	
dataOut >	::models::ReadData	0	> dataIn3	Display	
outDataAvailable >	::models::DataAvailable	20	> inDataAvailable	DisplayCorrelator	
modeChange >	::models::ChangeMode	0	> modeToggle1	PilotControl	
inDataAvailable <	::models::DataAvailable	20	< outDataAvailable	AirFrame	
dataIn <	::models::ReadData	20	< dataOut	AirFrame	
dataOut >	::models::ReadData	0	> dataIn2	Display	
timeOut <	::models::TimeOut	5	< timeOut5	EventChannel	
dataWriteOut <	::models::ReadWriteData	5	< dataWriteIn	NavSteeringPoints	
dataWriteIn >	::models::ReadWriteData	0	> dataWriteOut	Navigator	
outDataAvailable >	::models::DataAvailable	0	> inDataAvailableNavSteer...	NavSteering	
dataOut >	::models::ReadData	0	> dataIn1	NavSteering	

Aspect Synthesis

- Analysis algorithms
- Synthesis of QoS and real-time aspects

Aspect Synthesis

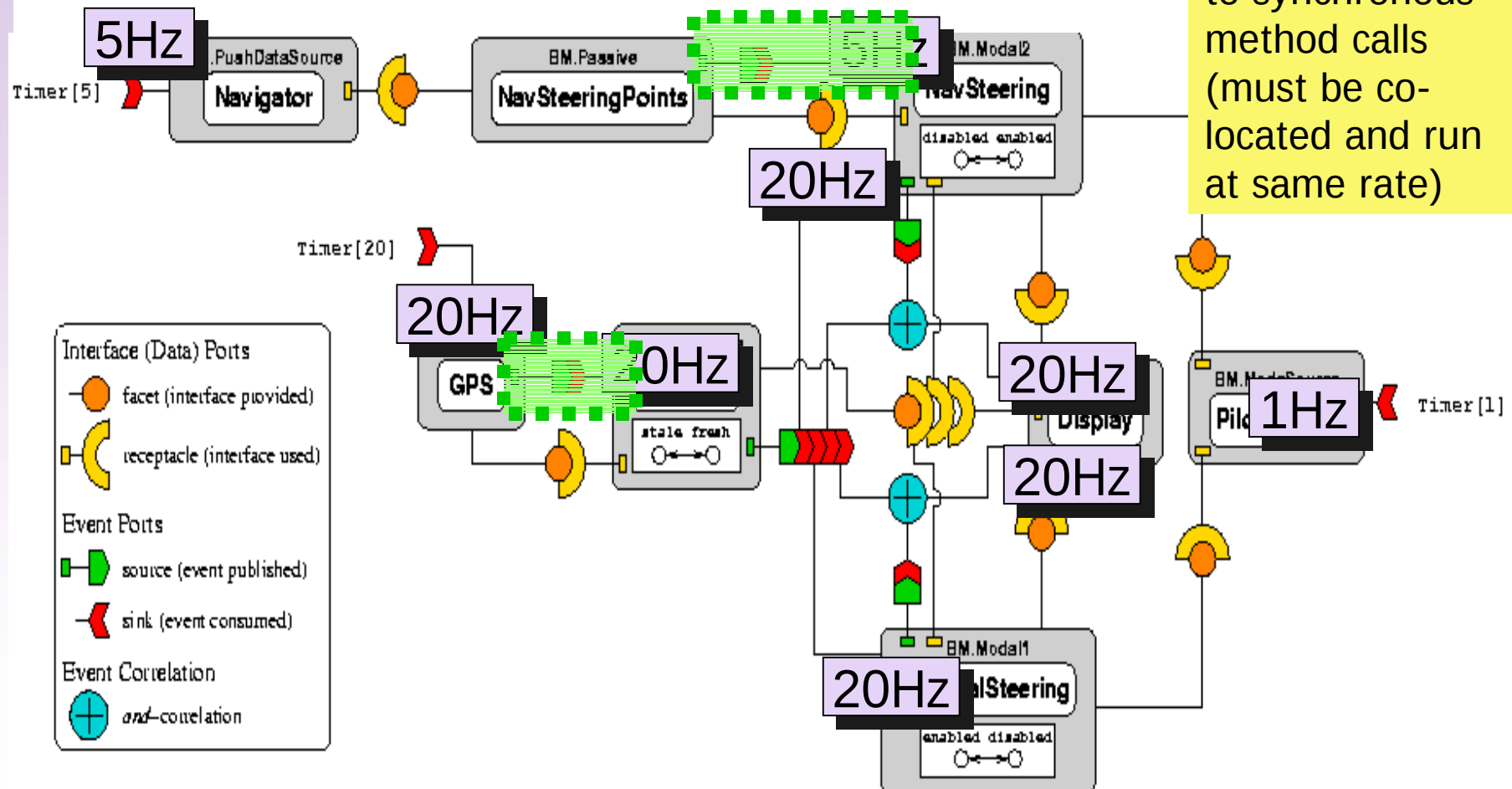
Dependency-driven rate assignment to event consumers



Aspect Synthesis

Automatic detection of optimization opportunities

Asynchronous message delivery to synchronous method calls (must be co-located and run at same rate)



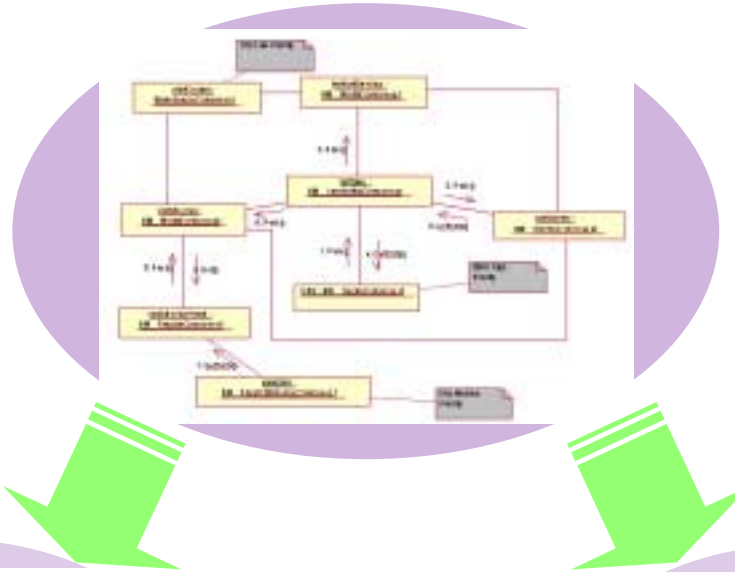
Spreadsheet View

Results of automatic rate group synthesis are fed back into spreadsheet

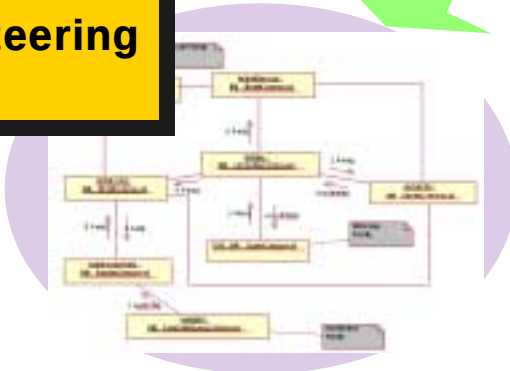
Rate (Hz)	Connected to	Contained in
5	> timeOut	Navigator
1	> timeOut	PilotControl
20	> timeOut	GPS
20	< timeOut2	EventChannel
20	> inDataAvailable	AirFrame
0	> dataIn	AirFrame
20	> inDataAvailable	TacticalSteering
20	> inDataAvailableAirFrame	DisplayCorrelator
20	> inDataAvailableAirFrame	NavSteering
20	< outDataAvailable	GPS
20	< dataOut	GPS
0	> dataIn	TacticalSteering
0	> dataIn1	Display
0	> dataIn2	NavSteering
20	< outDataAvailable	NavSteering
20	> inDataAvailable	Display
20	< outDataAvailable	AirFrame
20	< outDataAvailable	TacticalSteering
20	< dataOut	TacticalSteering
20	< dataOut	NavSteering
20	< outDataAvailable	DisplayCorrelator
20	< dataOut	AirFrame
1	< timeOut1	EventChannel
1	< modeChange	NavSteering
1	< modeChange	TacticalSteering
20	< dataOut	AirFrame
0	< outDataAvailable	NavSteeringPoints
20	> inDataAvailableNavSteer...	DisplayCorrelator
20	< outDataAvailable	AirFrame
0	< modeToggle1	PilotControl
20	< dataOut	NavSteeringPoints
0	> dataIn3	Display
20	> inDataAvailable	DisplayCorrelator
0	< modeToggle1	PilotControl
20	< outDataAvailable	AirFrame
20	> dataOut	AirFrame
0	> dataIn2	Display
5	< timeOut5	EventChannel
5	< dataWriteIn	NavSteeringPoints
0	> dataWriteOut	Navigator
0	> inDataAvailableNavSteer...	NavSteering
0	> dataIn1	NavSteering

Mode-based Projections

Scenario Diagram w/
Complete Connectivity

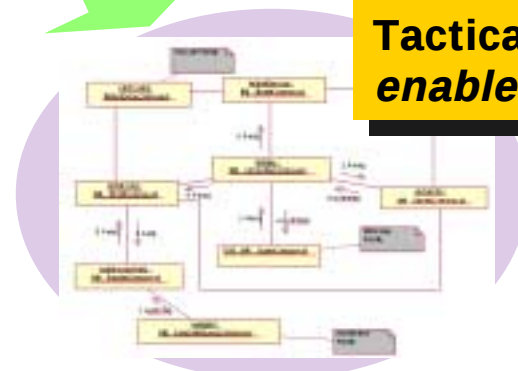


**NavSteering
enabled
TacticalSteering
*disabled***



■ ■ ■

**NavSteering
disabled
TacticalSteering
*enabled***



Enabled Connectivity for Different Modes

Mode-based Projections

The screenshot shows a software development environment with a project browser on the left, a workspace in the center, and an outline on the right. A dialog box is open in the center, displaying a table of mode variables and their values. The table has two columns: 'Mode Variable' and 'Values'. The variables listed are 'AirFrame#LazyActiveMode', 'TacticalSteering#OnOffMode', and 'NavSteering#OnOffMode'. The values are 'stale', 'enabled', and 'enabled' respectively. The 'NavSteering#OnOffMode' row is highlighted, and a dropdown menu is open showing 'enabled' and 'disabled' options. A yellow callout box points to the dropdown menu with the text '...possible values for mode variables'.

Mode Variable	Values
AirFrame#LazyActiveMode	stale
TacticalSteering#OnOffMode	enabled
NavSteering#OnOffMode	enabled

...possible values for mode variables

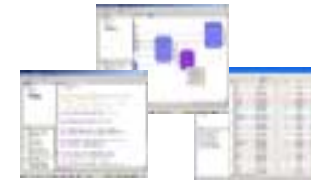
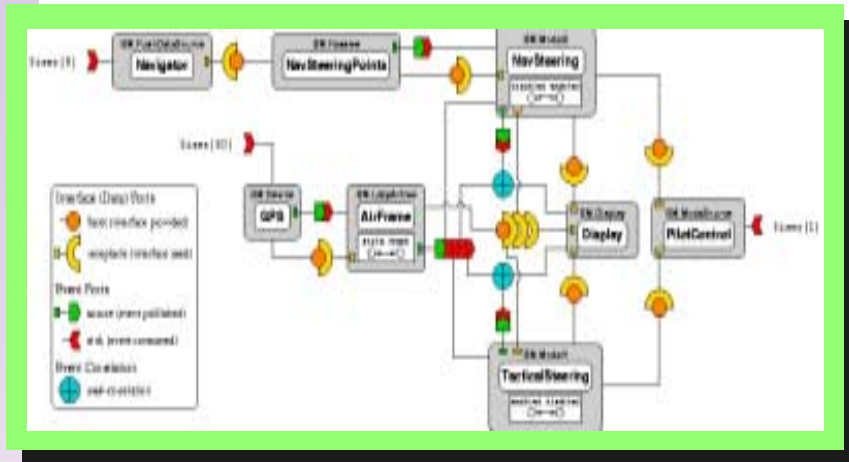
OK Cancel

Multiple *mode-based views* are automatically created and synchronized through the design process.

Model-Checking Infrastructure

- Infrastructure dedicated to event-based middleware
- Complete model derived from CCM IDL, component-assembly description and annotations
- Model-checking for global system properties

Complete Model



Configuration Info

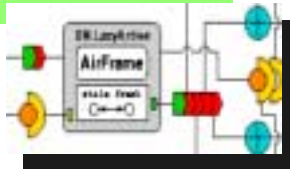
Transition
System
Specs

System Components

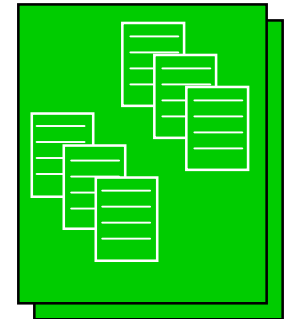
RT CORBA Event Channel

Leverage CO R B A IDL

```
component BMLazyActive {  
  provides ReadData outData;  
  uses ReadData inData;  
  publishes DataAvailable outDataAvailable;  
  consumes DataAvailable inDataAvailable;  
  attribute LazyActiveMode dataStatus;  
};
```



IDL Compiler



Component
Implementation
Stubs & Skeletons

+

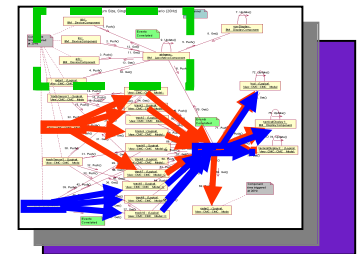
```
dependencydefault  
  == none;  
dependencies {  
  inDataAvailable  
  ->  
  outDataAvailable;  
}
```

Dependency
Annotations

```
behavior {  
  if (mode==enabled) {  
    push outDataAvailable;  
  }  
  ...  
}
```

Transition System
Semantics

Model Builder



Dependency Analysis
and
Model-checking Engine

Incremental Specification

Specifications

Component Structure

*Increasing
Effort
&
Strength of Verification*

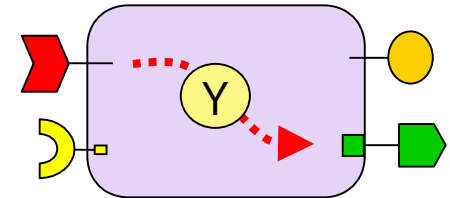
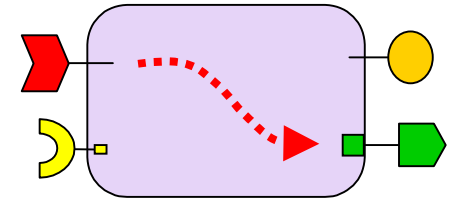
port action
dependencies

refinement

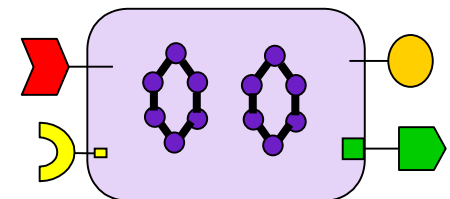
state-based
dependencies

refinement

component transition
semantics



...only in mode Y

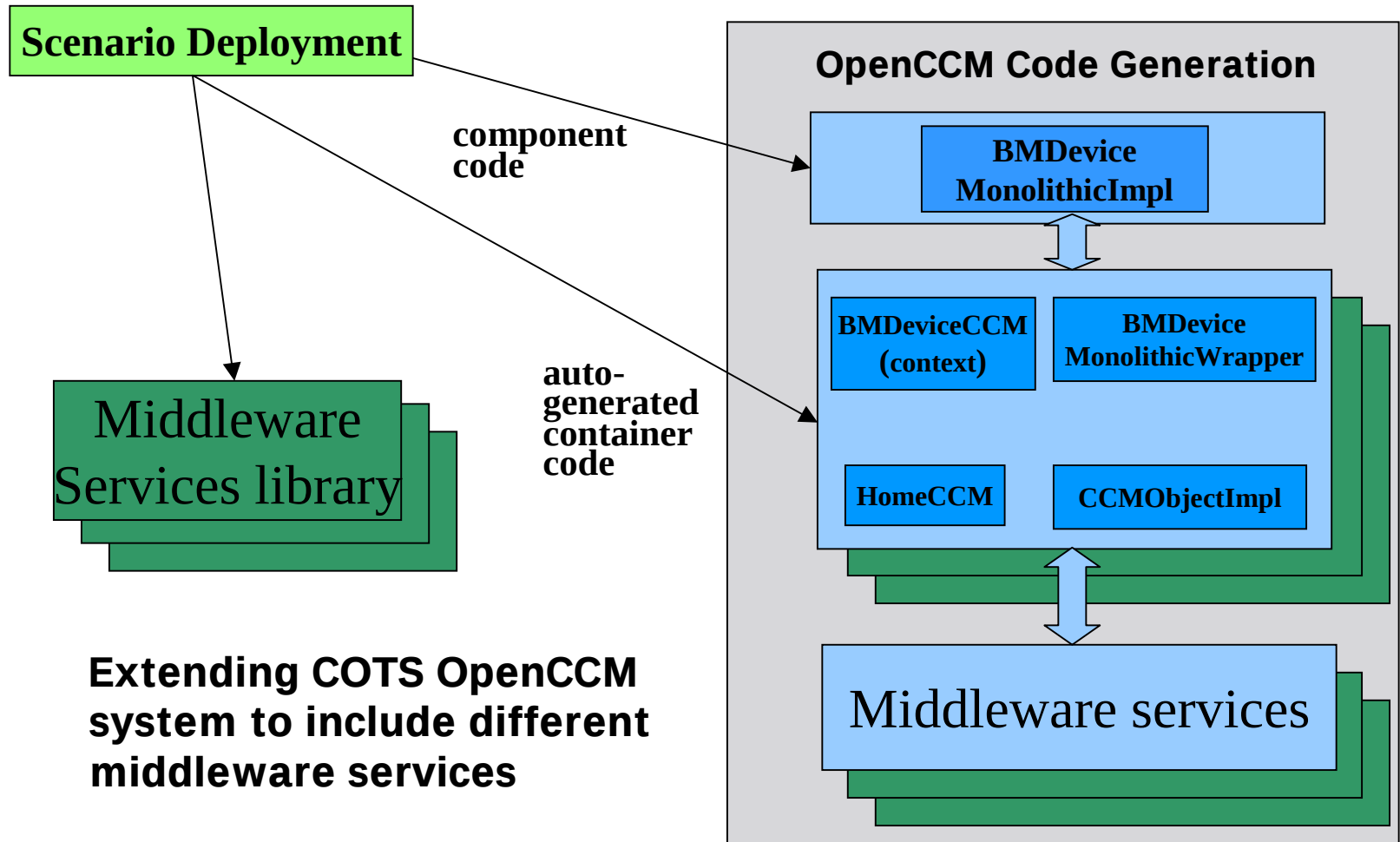


*...state machines give
abstract behavior*

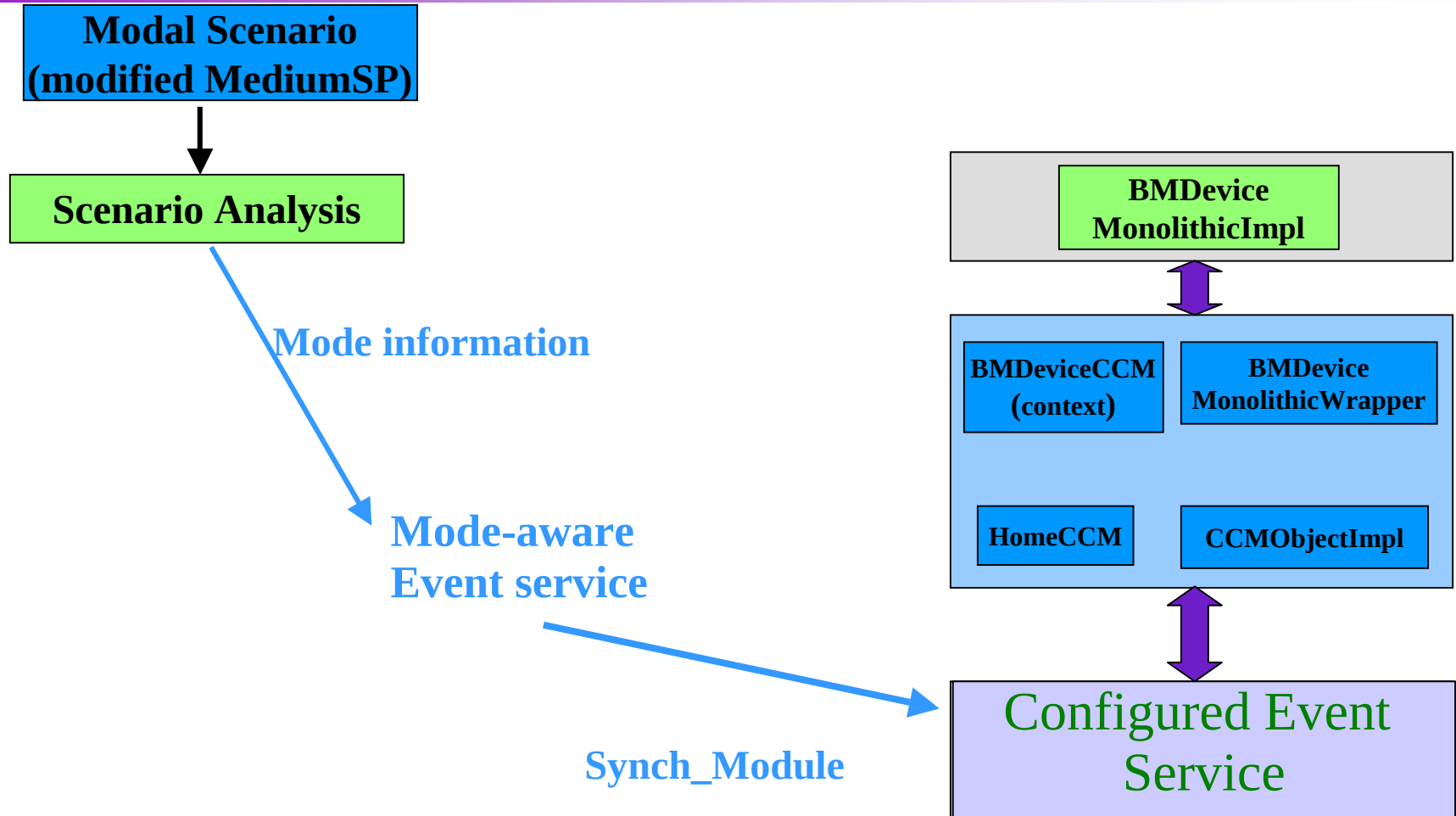
Code generation and Configuration

- OpenCCM CCM IDL to Java compiler to component stub and skeleton code generation
- XML files to container and middleware configuration

Container Adaptation



Experiments: Mode-aware Event Service



Conclusions

- Cadena: model-driven design and implementation of DRE systems
- Light-weight specifications
- Component assembly description forms
- Dependency analysis tools
- Model-check infrastructure
- Code generation and configuration

Project Web Site

<http://www.cis.ksu.edu/cadena>