How to Harvest Reusable Components in Existing Software

Nikolai Mansurov

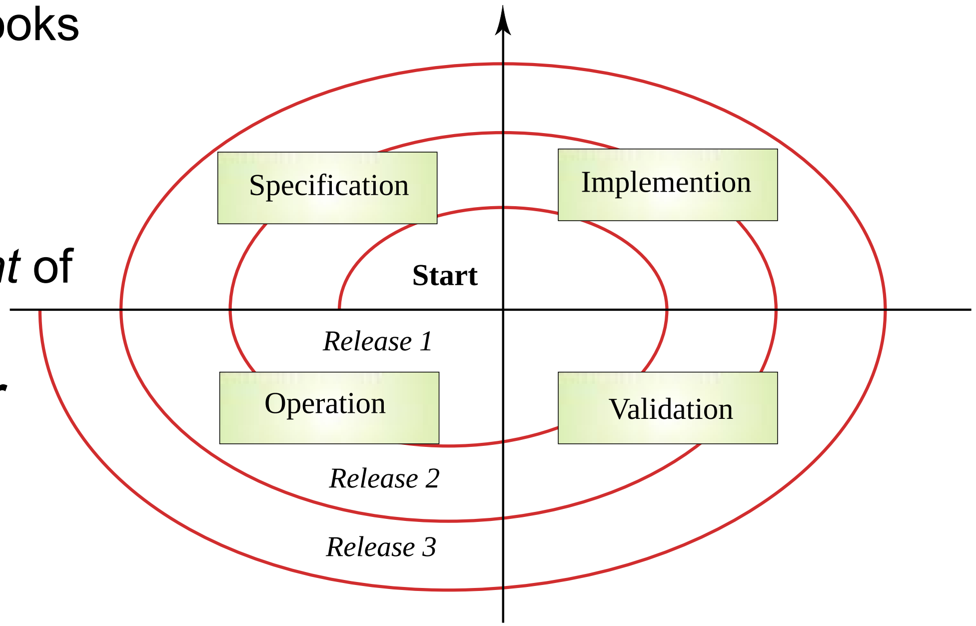
Chief Scientist & Architect

Overview

- **Introduction**
- **Reuse, Architecture and MDA**
- **Option Analysis for Reengineering (OAR)**
- **Architecture Excavation and Refactoring**
- **Harvesting as architectural refactoring**
  - First, you need to excavate your architecture, as a precise model
  - Harvesting requires the following steps:
    - Identify/localize/collect
    - Isolate component
    - Introduce a boundary
    - Implement refactoring
  - Particular challenges
    - Challenges of component identification (need to understand)
    - Challenges of component isolation
    - Challenges of soft boundaries
    - Reusable components and architecture erosion
    - Reusable components and cyclic dependencies
- **Examples**
- **Conclusions/Key points**

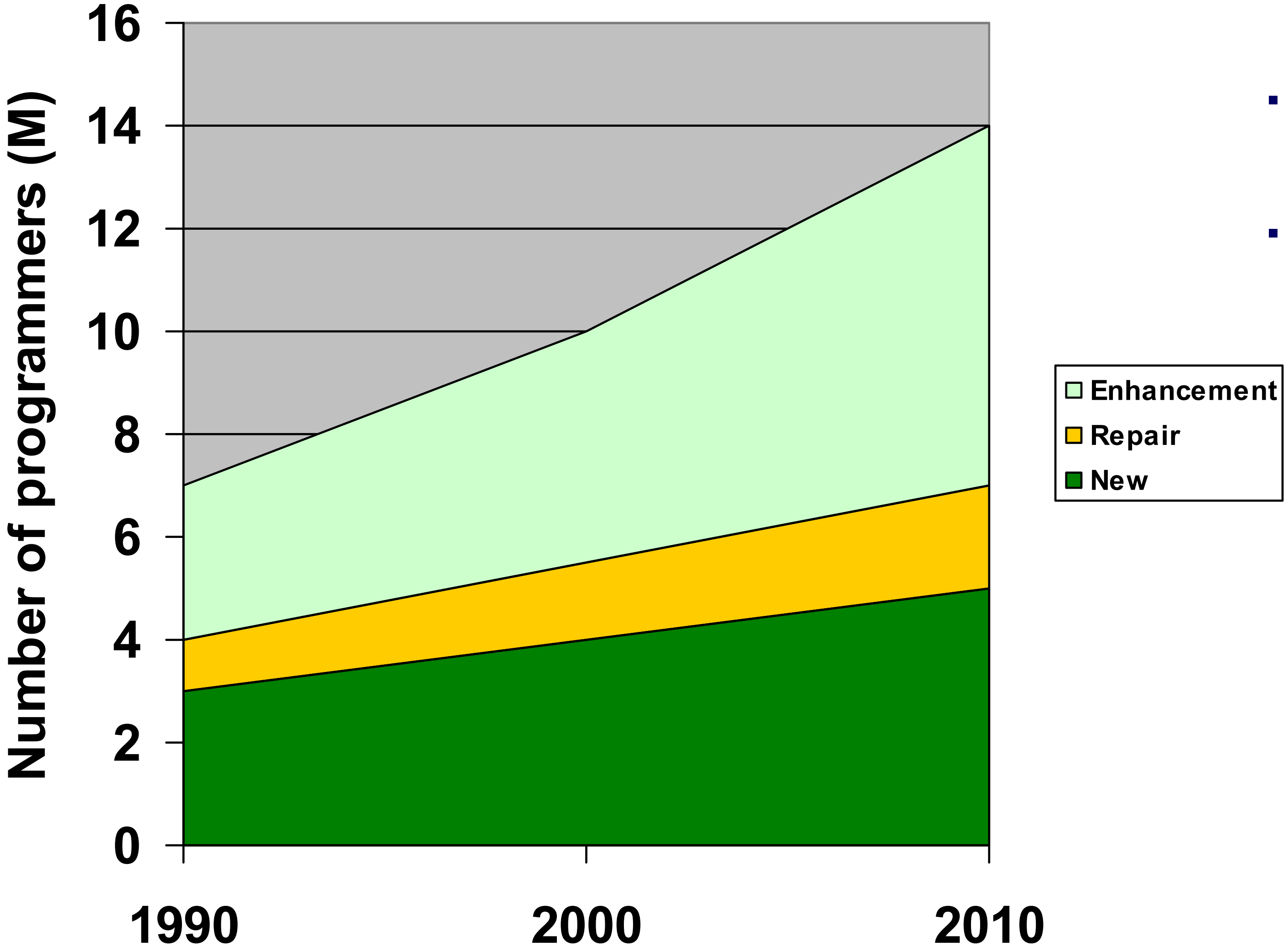
Production of software is evolutionary

- **Production of software involves *multiple* releases**
  - …despite the fact that many textbooks emphasize only the *initial* release, when *the system is built*
  - Software production after the initial release has an additional *constraint* of dealing with *existing code*
- **Changes are made to software after the initial release in order to:**
  - Develop new functionality
  - Fix bugs
  - Adapt to new operating environments
  - Improve quality



From Ian Sommerville, Software Engineering, 2000

Evolution of existing code has significant economic impact

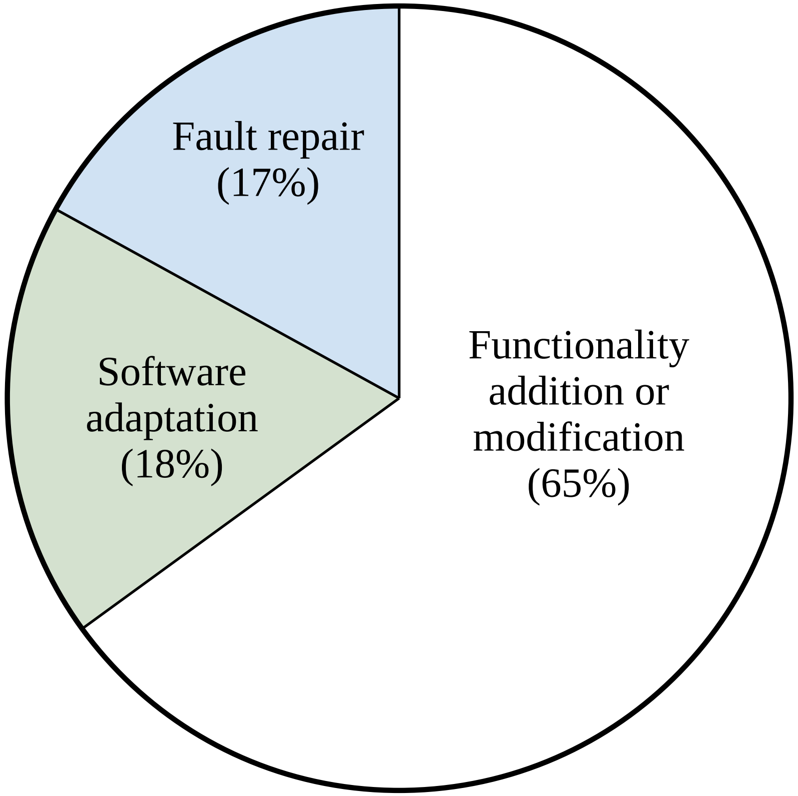


From Deursen,Klint,Verhoef,1999

- **More than one half of all programmers are *already* working with existing code (repair + enhancement)**
- **The cost of post-initial evolution has grown from 40% to 90% of the total production cost**
- **The number of programmers working with existing code *grows faster* than the number of programmers developing new software**
  - Larger amounts of software already developed
  - Large business value accumulates in existing code over time
  - Increasing cost of new development
  - The useful lifespan is increasing
  - Bigger churn of platforms and technologies
  - More defects

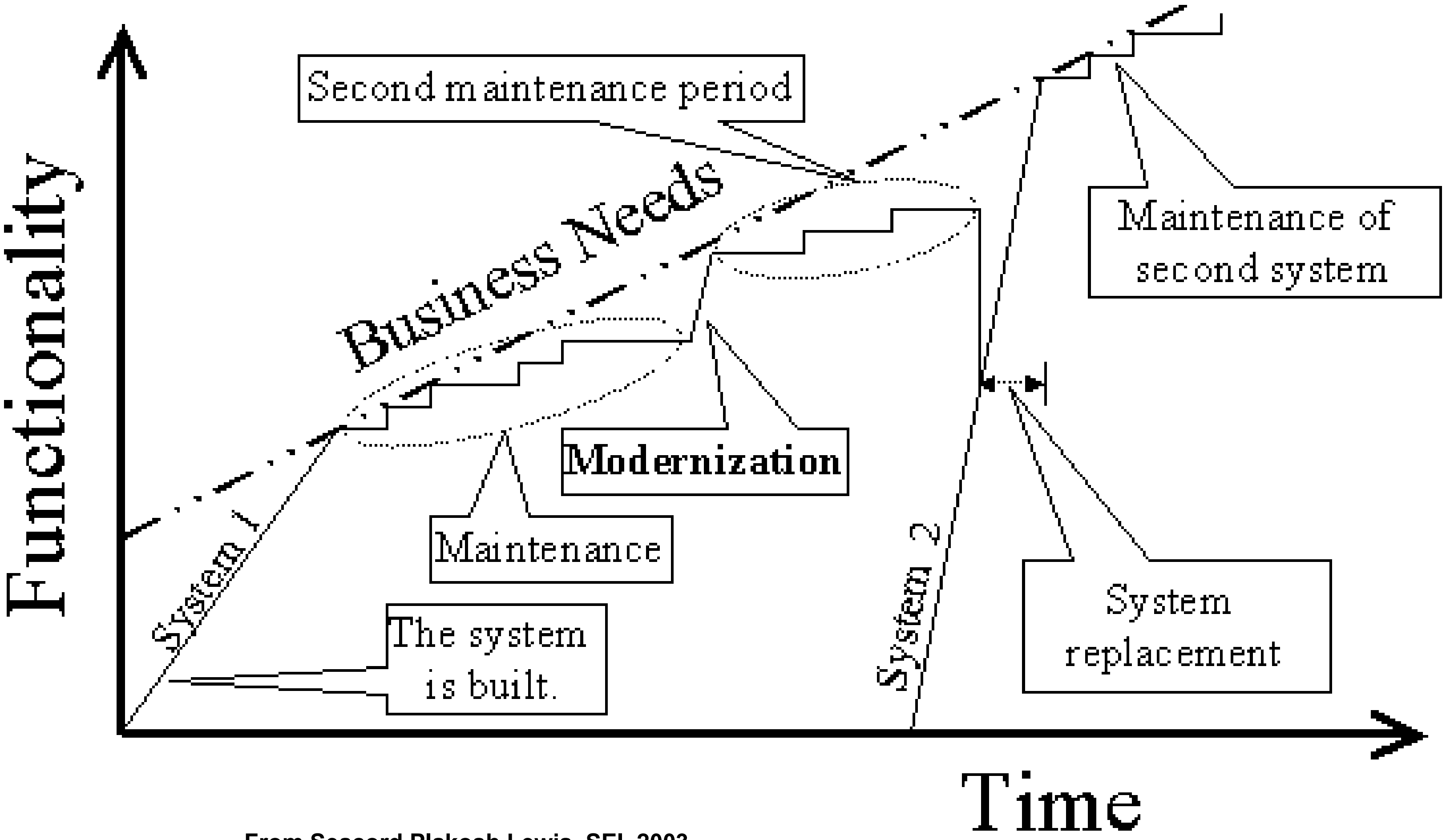
Software production with existing code

- **Changes to existing code may have different *magnitude:***
  - Traditional maintenance (repair)
    - Perfective maintenance
    - Adaptive maintenance
    - Corrective maintenance
    - Preventive maintenance
  - Major new features to existing software
    - Major modifications
    - Scaling
  - Modernization (beyond maintenance)
    - Porting to a new platform
    - Migration to a new technology
    - Migration to COTS components
    - Modularization and refactoring
  - Redevelopment



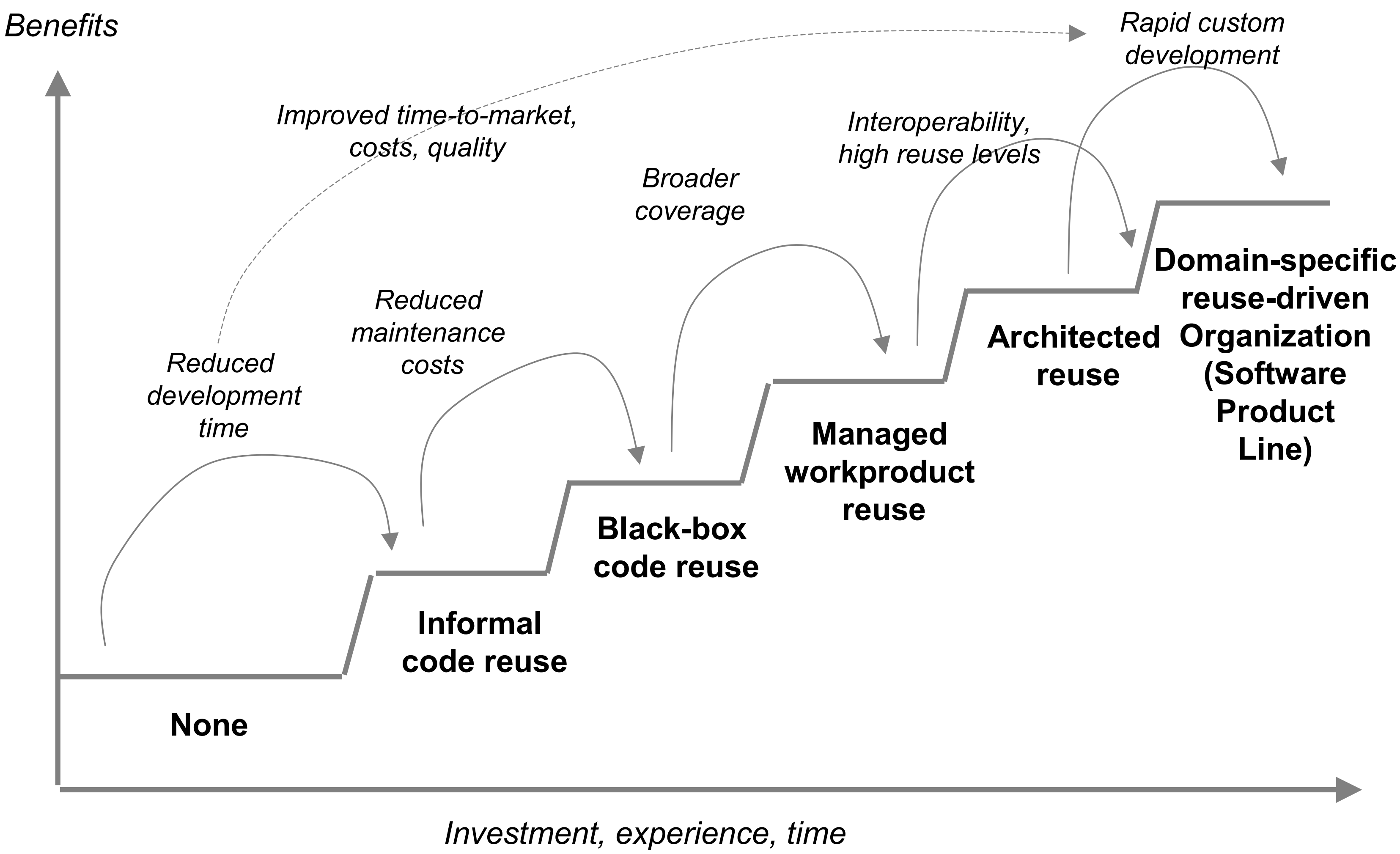
From Ian Sommerville, Software Engineering, 2000

Changes to existing software involve *reuse*



From Seacord,Plakosh,Lewis, SEI, 2003

The levels of reuse

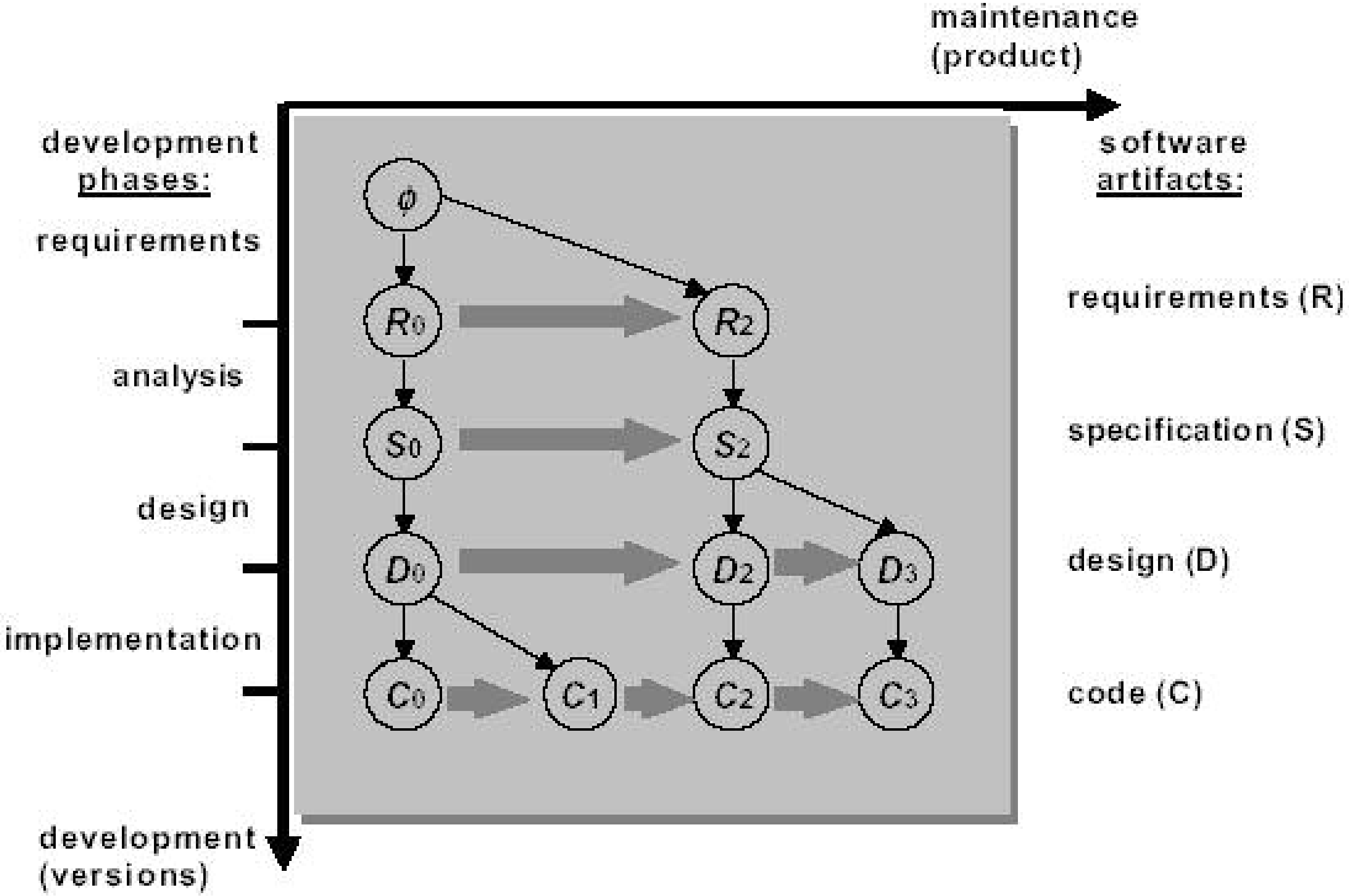


Adapted from Jacobson, et.al., 1997

Kinds of reuse

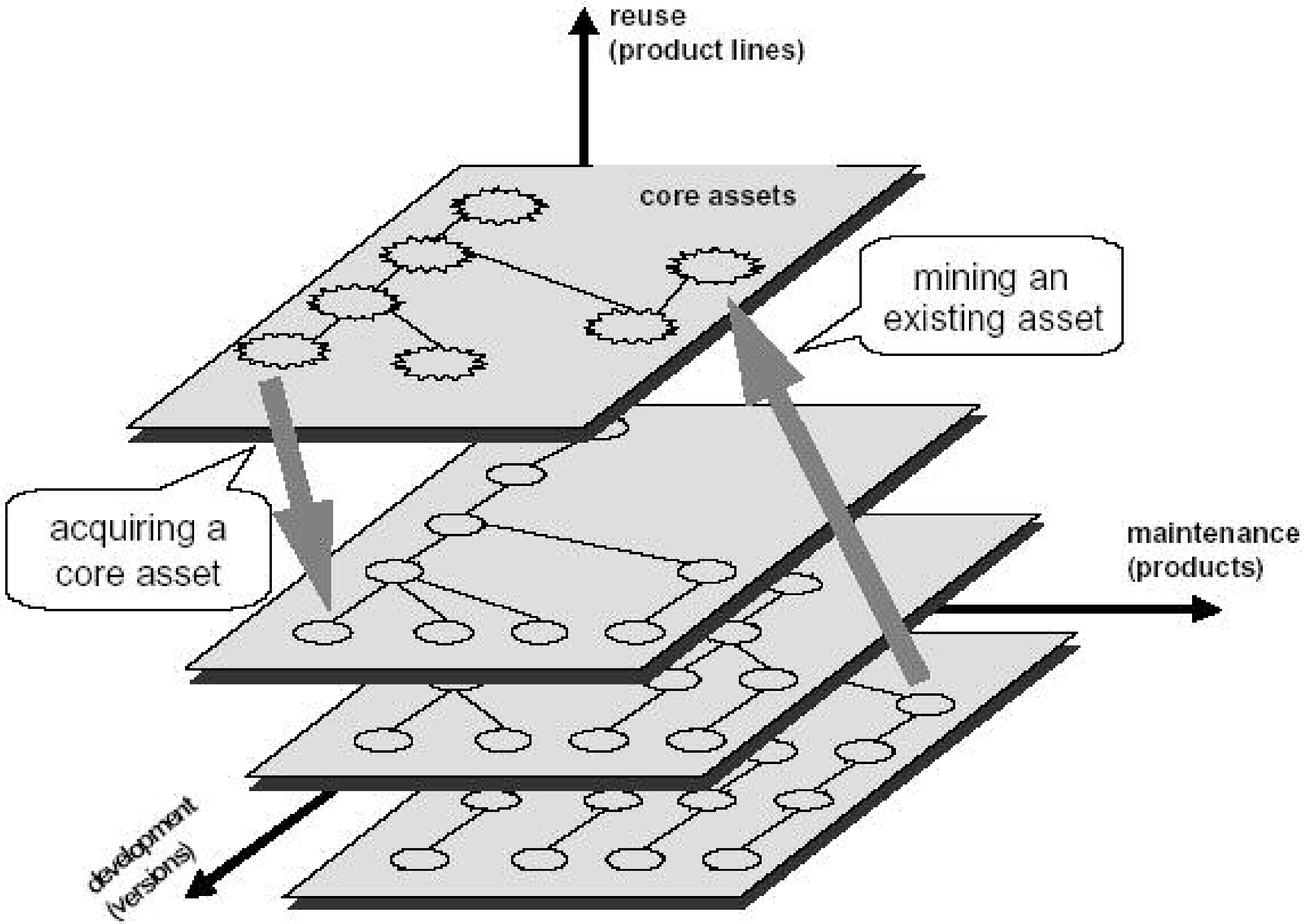
- "As is" reuse
- "Black-box" reuse
- "White box" reuse
- Workproduct reuse

Reuse in multi-release production



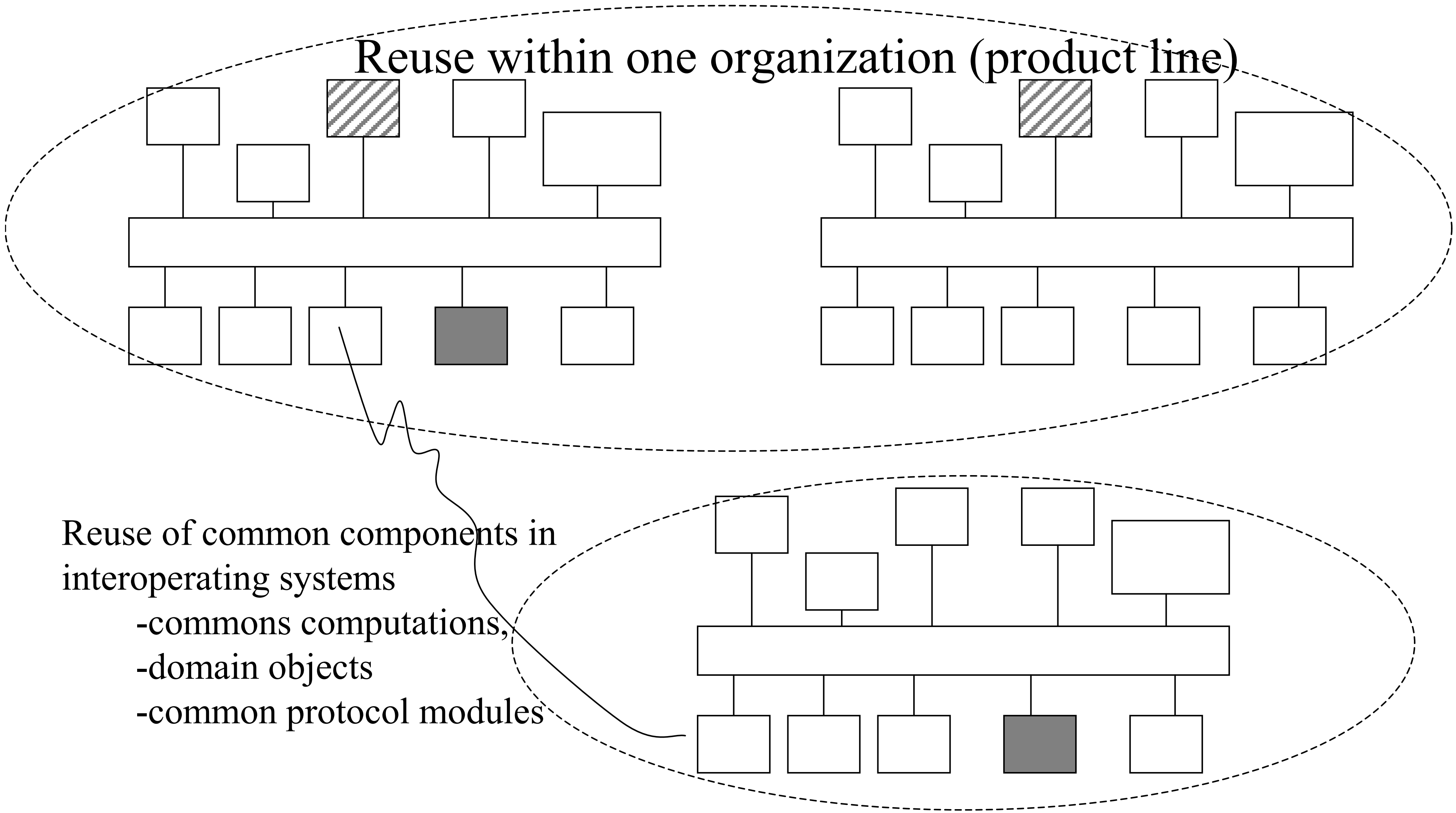
From Schach,Tomer, 2001

Product Lines add another dimension to production



From Schach,Tomer, 2001

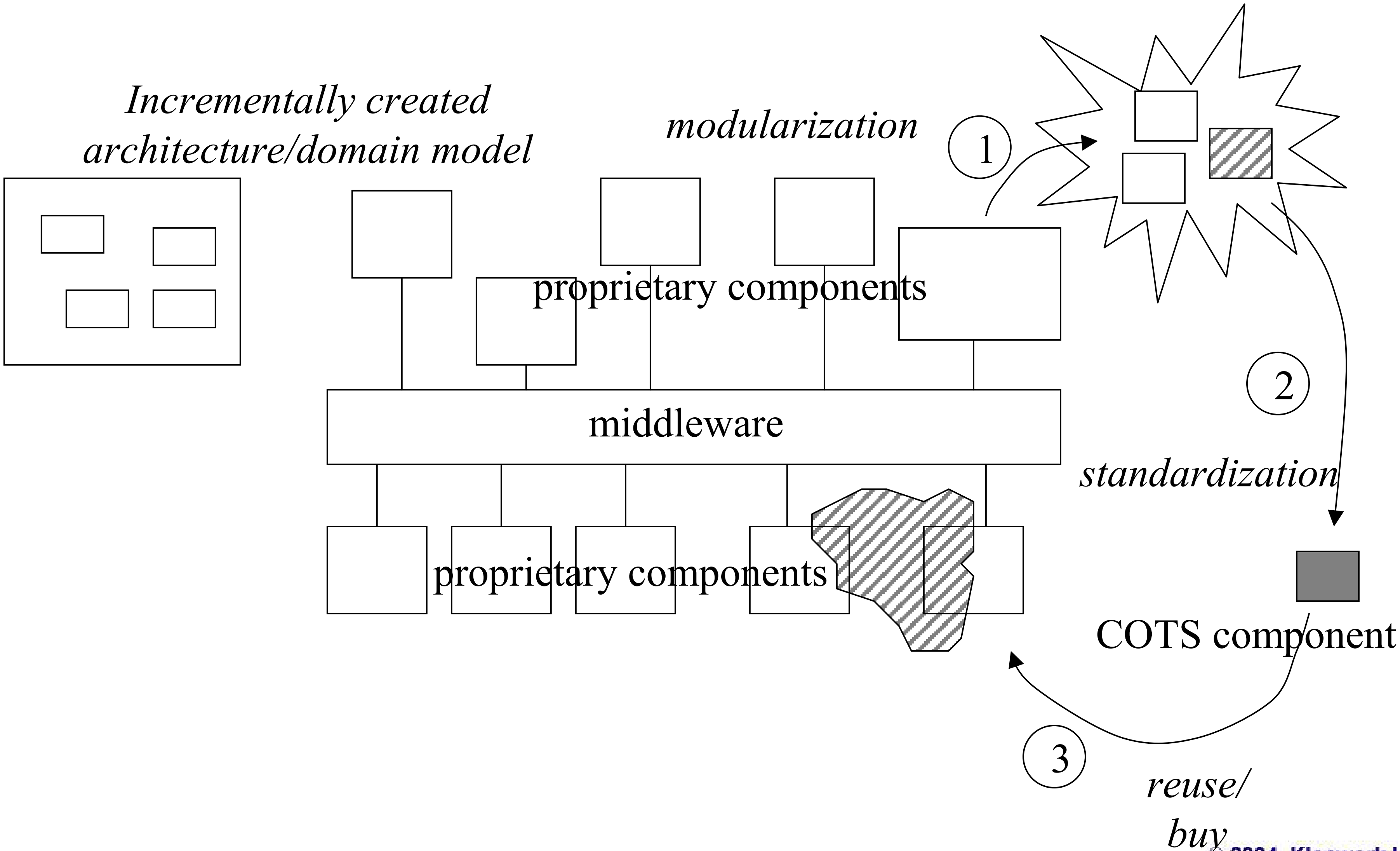
Internal and External reuse



Reuse of common components in interoperating systems

- -commons computations,
- -domain objects
- -common protocol modules

Reuse and COTS components



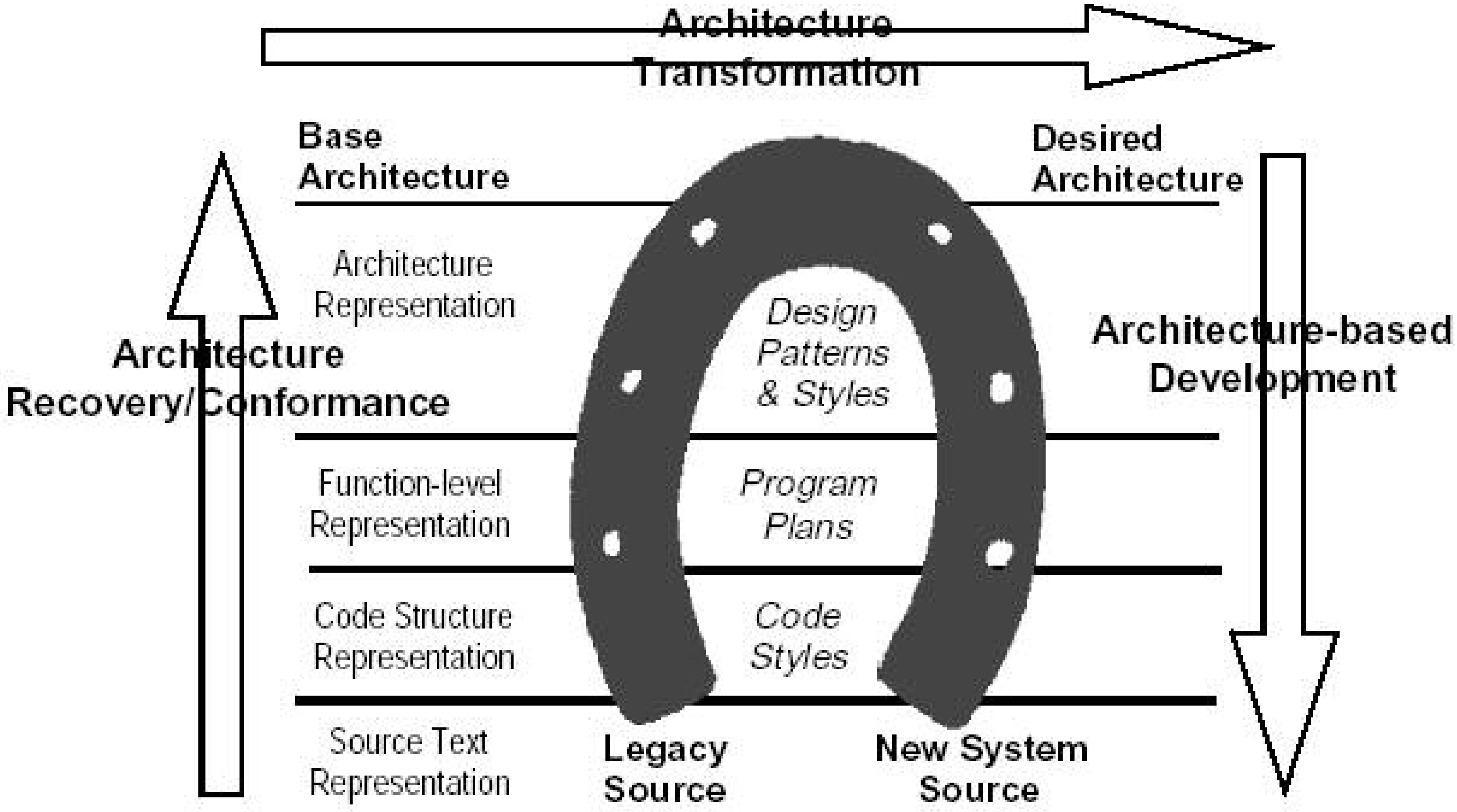
Harvesting, standardization, reuse

- (Porting to standard COTS middleware)
- Modularization
- Harvesting of reusable components
- Standardization of components
- Reuse/buy
- Emergent architecture/domain model

Overview

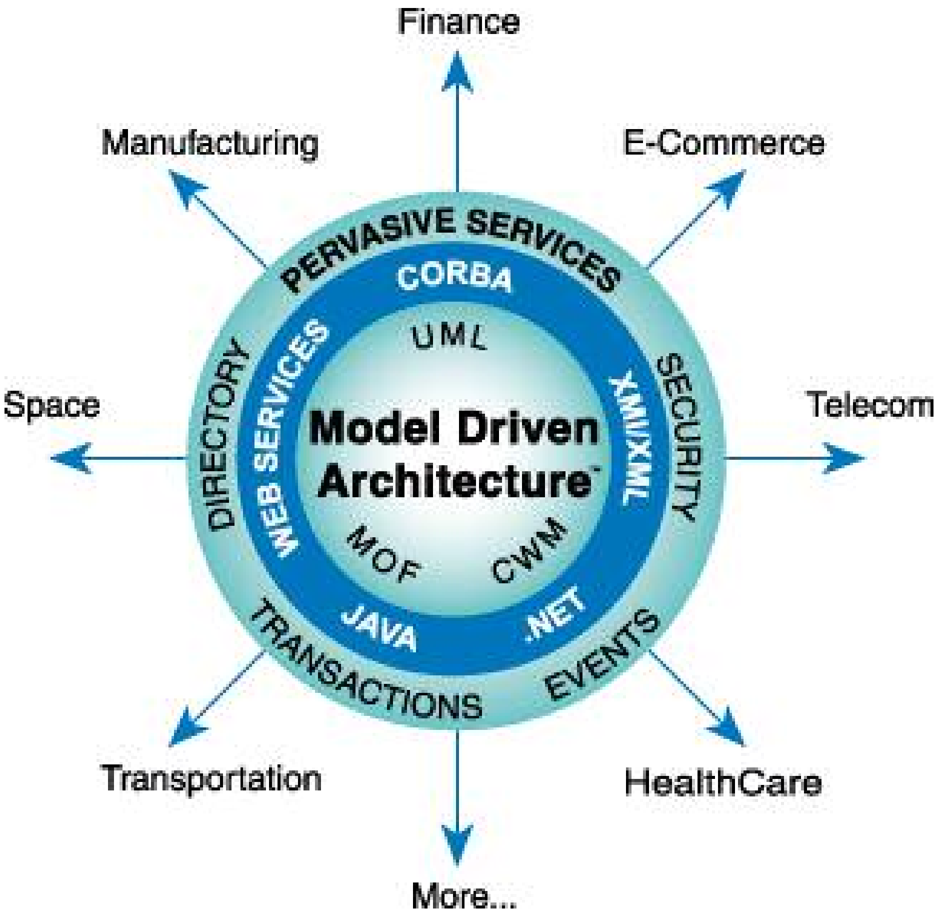
- **Reuse, Architecture and MDA**
  - Reuse does not involve any translation
  - What can be reused as opposed to what can be translated
    - reuse at low levels and at higher levels

The SEI HorseShoe model



From Smith, et.al., SEI, 1999

Model Driven Architecture



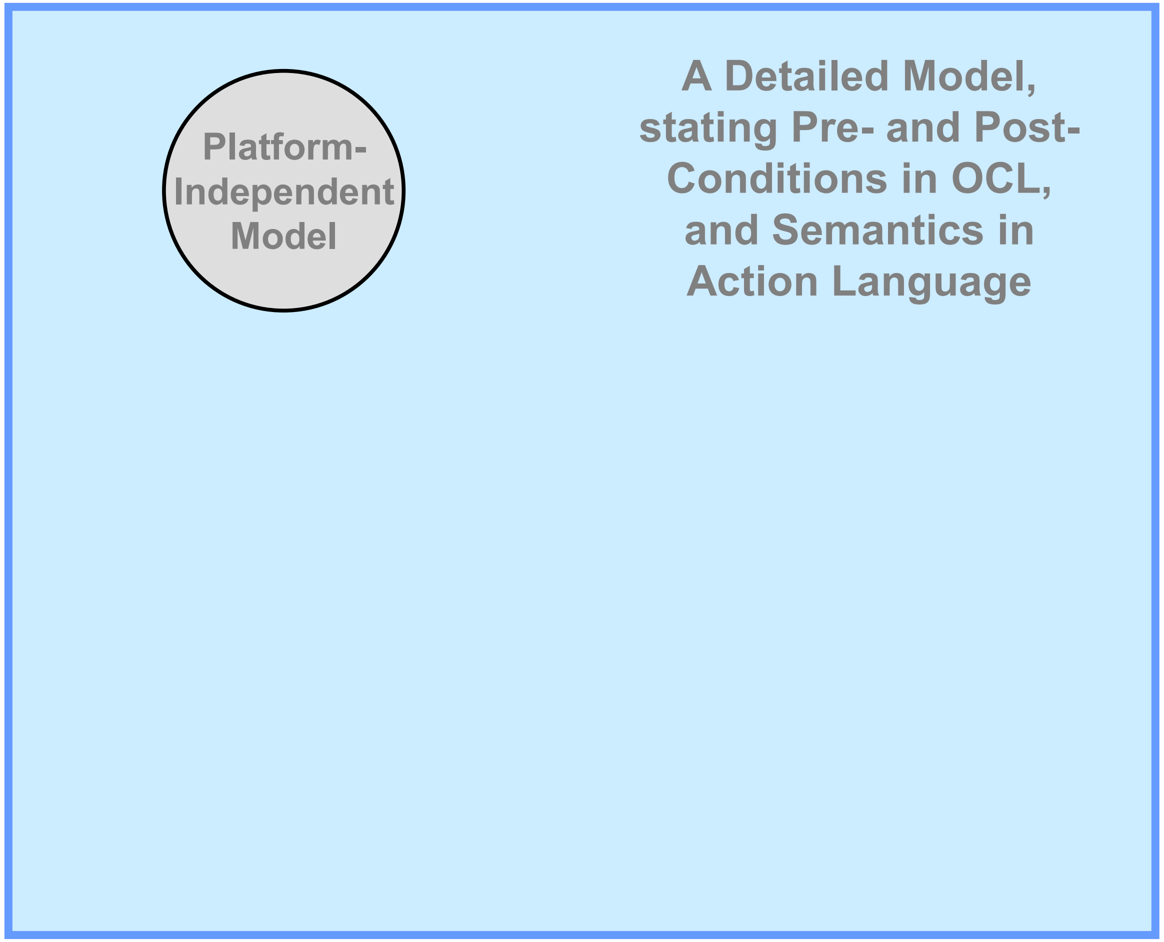
From R.Soley, OMG, 2003

What is Model Driven Architecture?

- **A New Way to Specify and Build Systems**
  - ***Based on modeling with UML***
  - Supports full lifecycle: analysis, design, implementation, deployment, maintenance, evolution & integration with later systems
  - Builds in Interoperability and Portability
  - Lowers initial cost and maximizes ROI
  - Applies directly to the mix you face:
    - Programming language
    - Operating system
    - Network
    - Middleware

From R.Soley, OMG, 2003

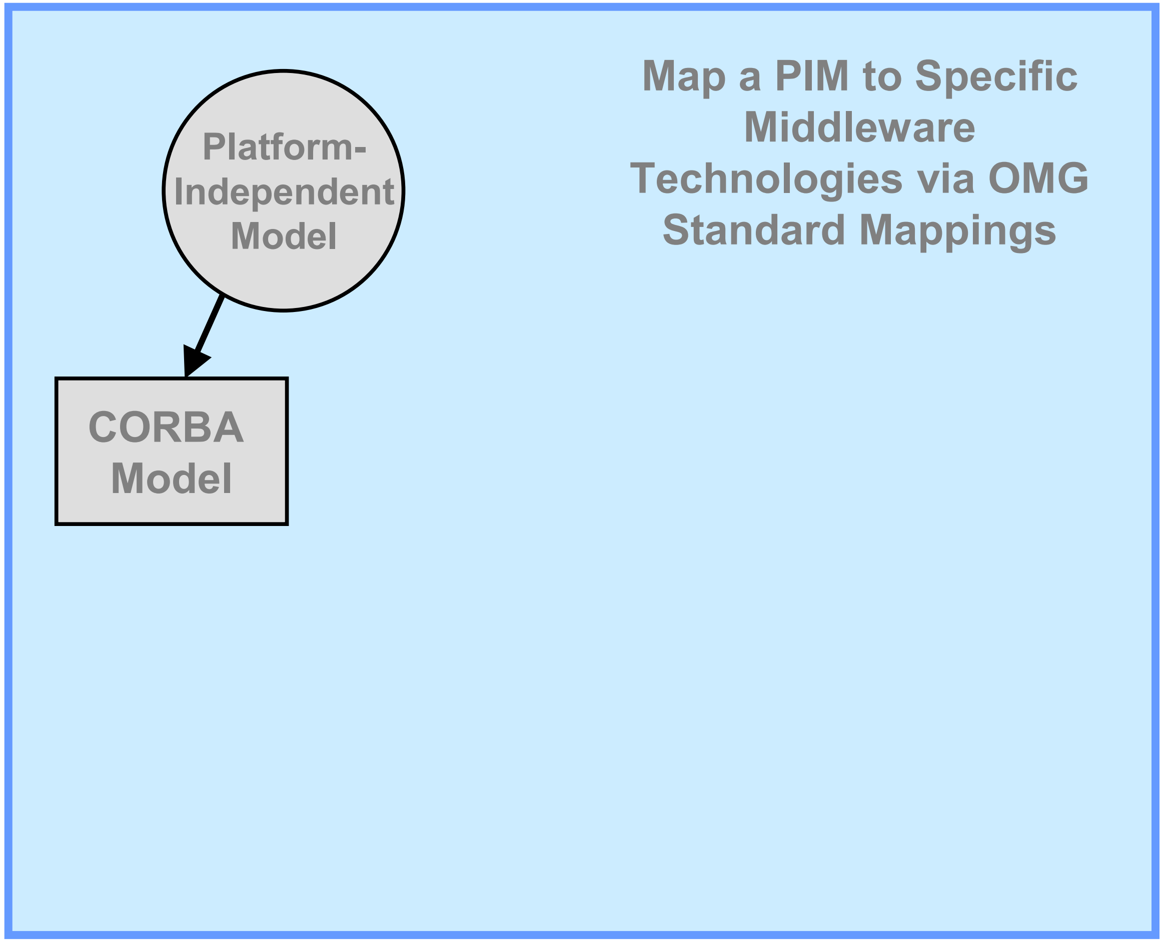
Building an MDA Application



Start with a *Platform-Independent Model* (PIM) representing business functionality and behavior, undistorted by technology details.

From R.Soley, OMG, 2003

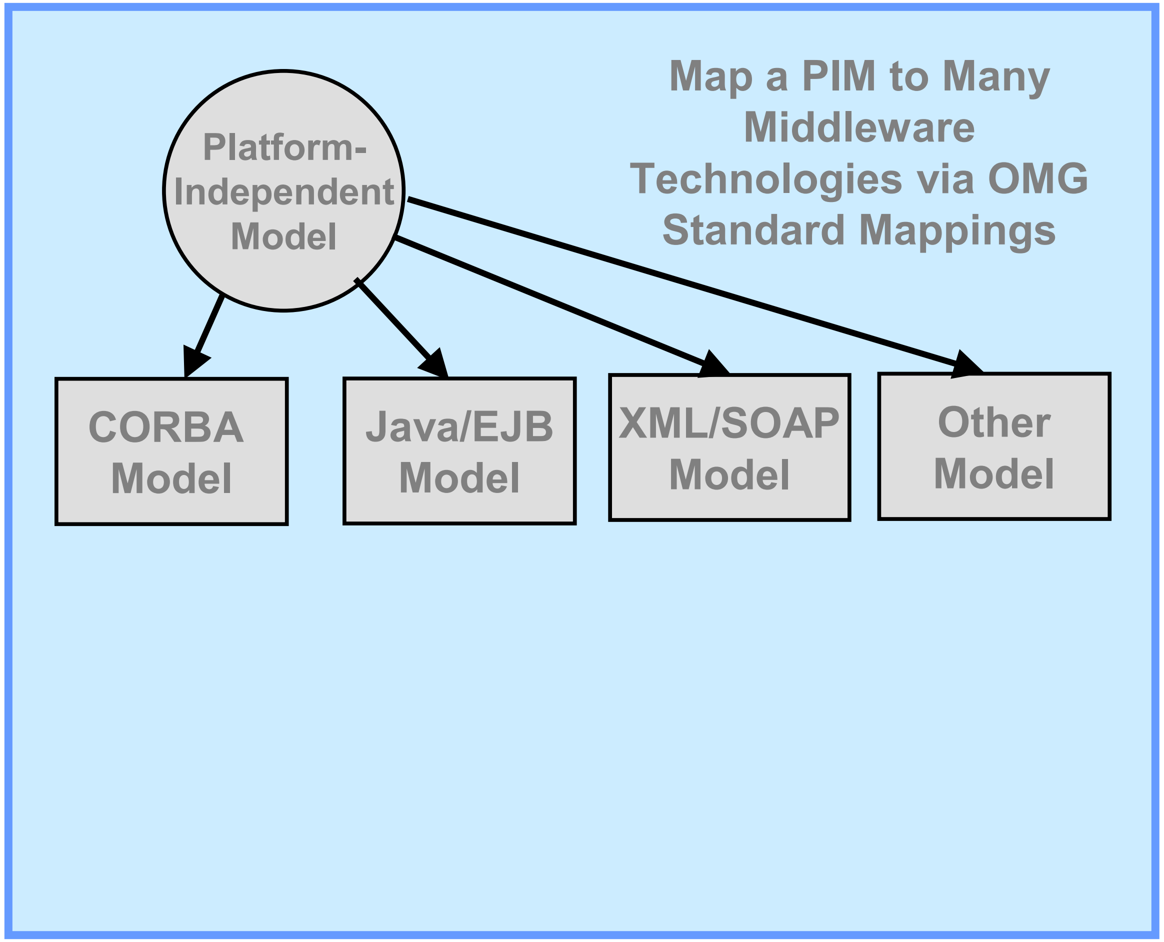
Generating Platform-Specific Model



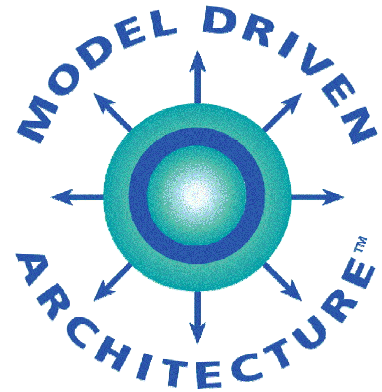
MDA tool applies a standard mapping to generate *Platform-Specific Model* (PSM) from the PIM. Code is partially automatic, partially hand-written.

From R.Soley, OMG, 2003

Mapping to Multiple Deployment Technologies

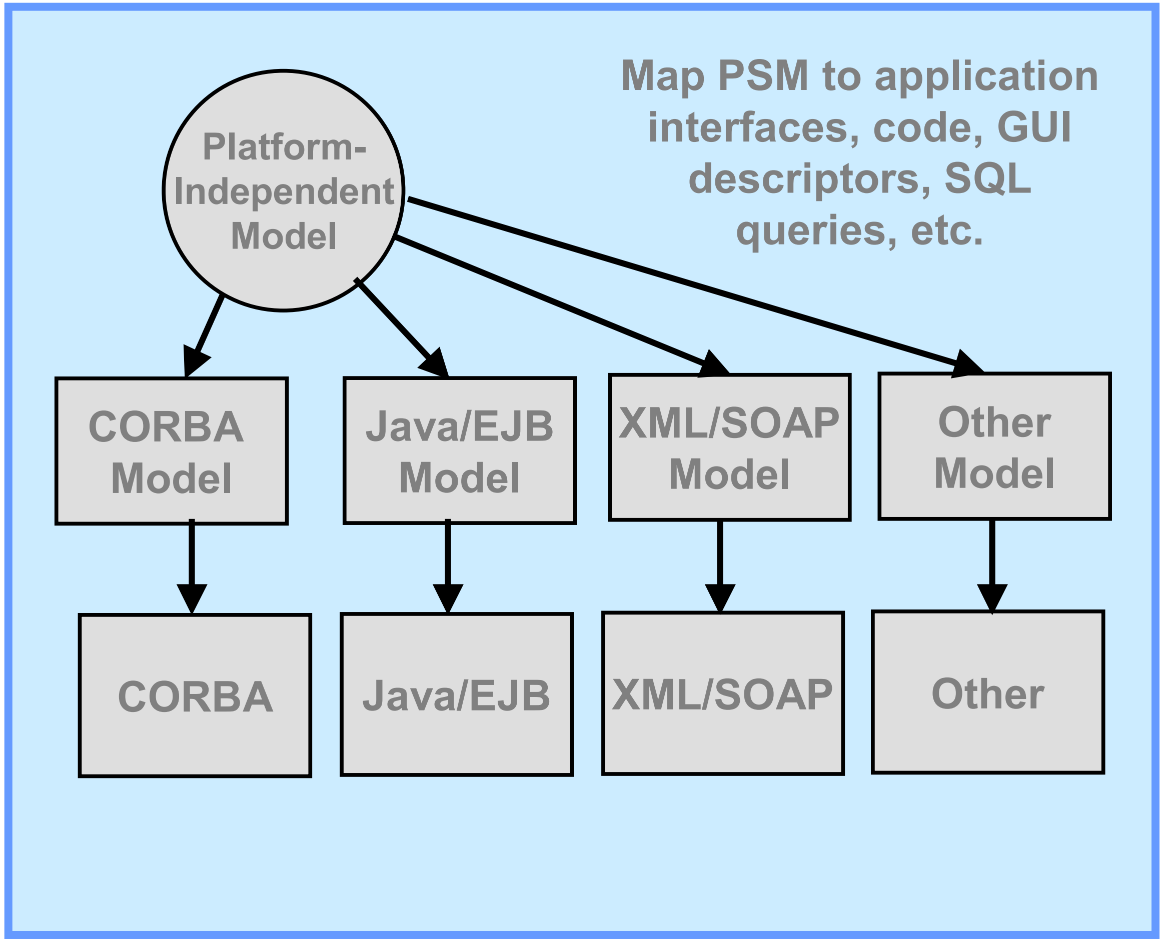


MDA tool applies an standard mapping to generate *Platform-Specific Model* (PSM) from the PIM. Code is partially automatic, partially hand-written.



From R.Soley, OMG, 2003

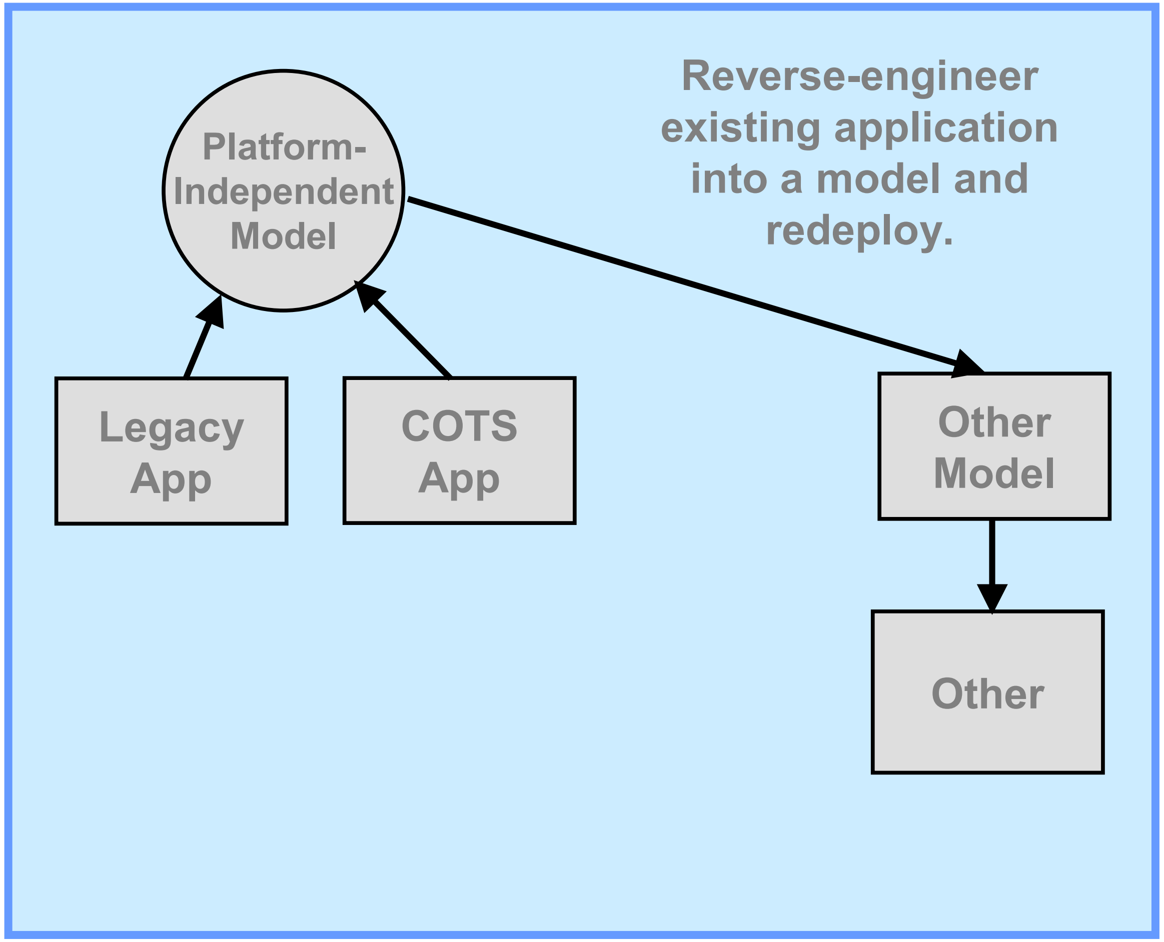
Generating Implementations



MDA Tool generates all or most of the implementation code for deployment technology selected by the developer.

From R.Soley, OMG, 2003

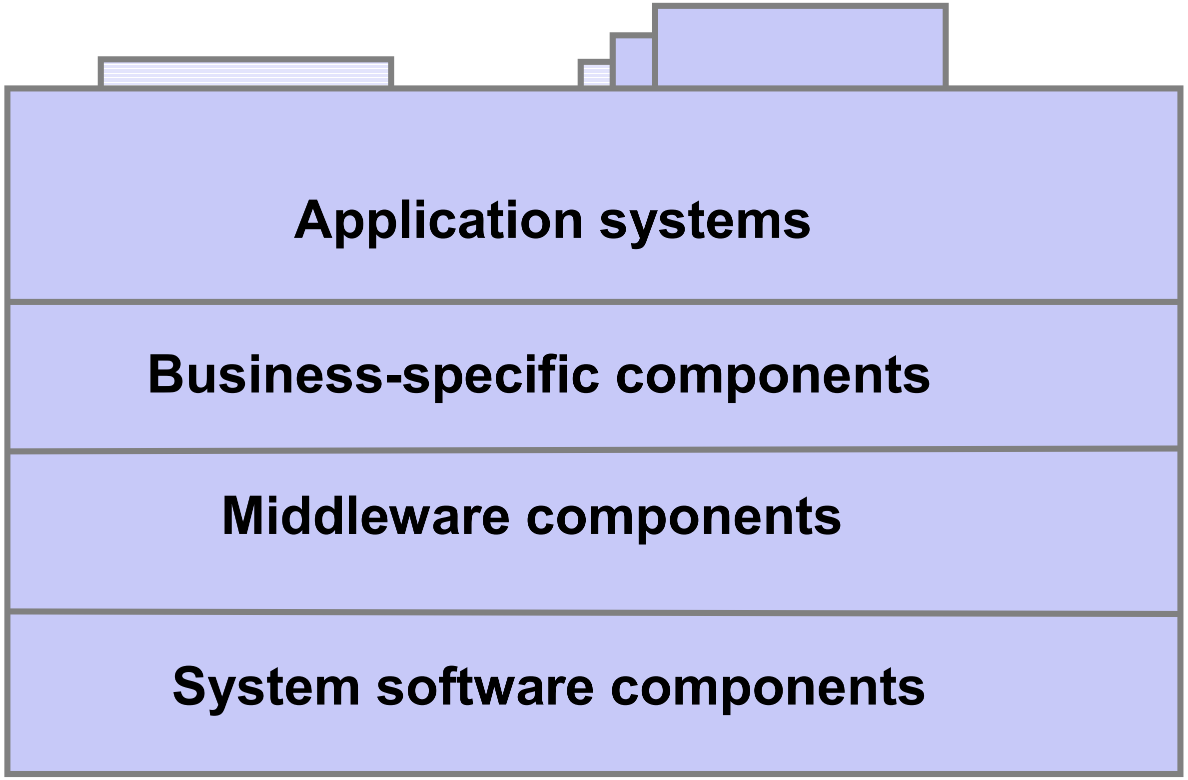
Integrating Legacy & COTS



MDA Tools for reverse engineering automate discovery of models for re-integration on new platforms.

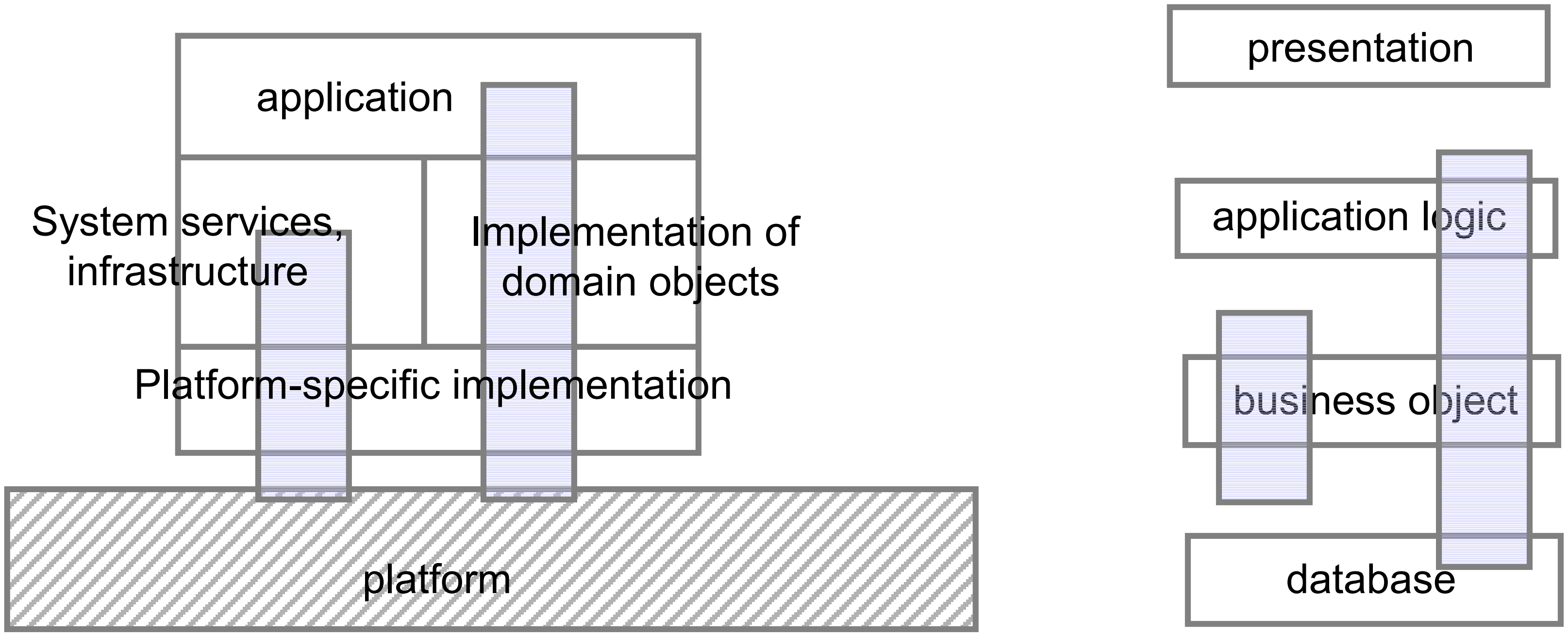
From R.Soley, OMG, 2003

A typical architecture for Business Reuse

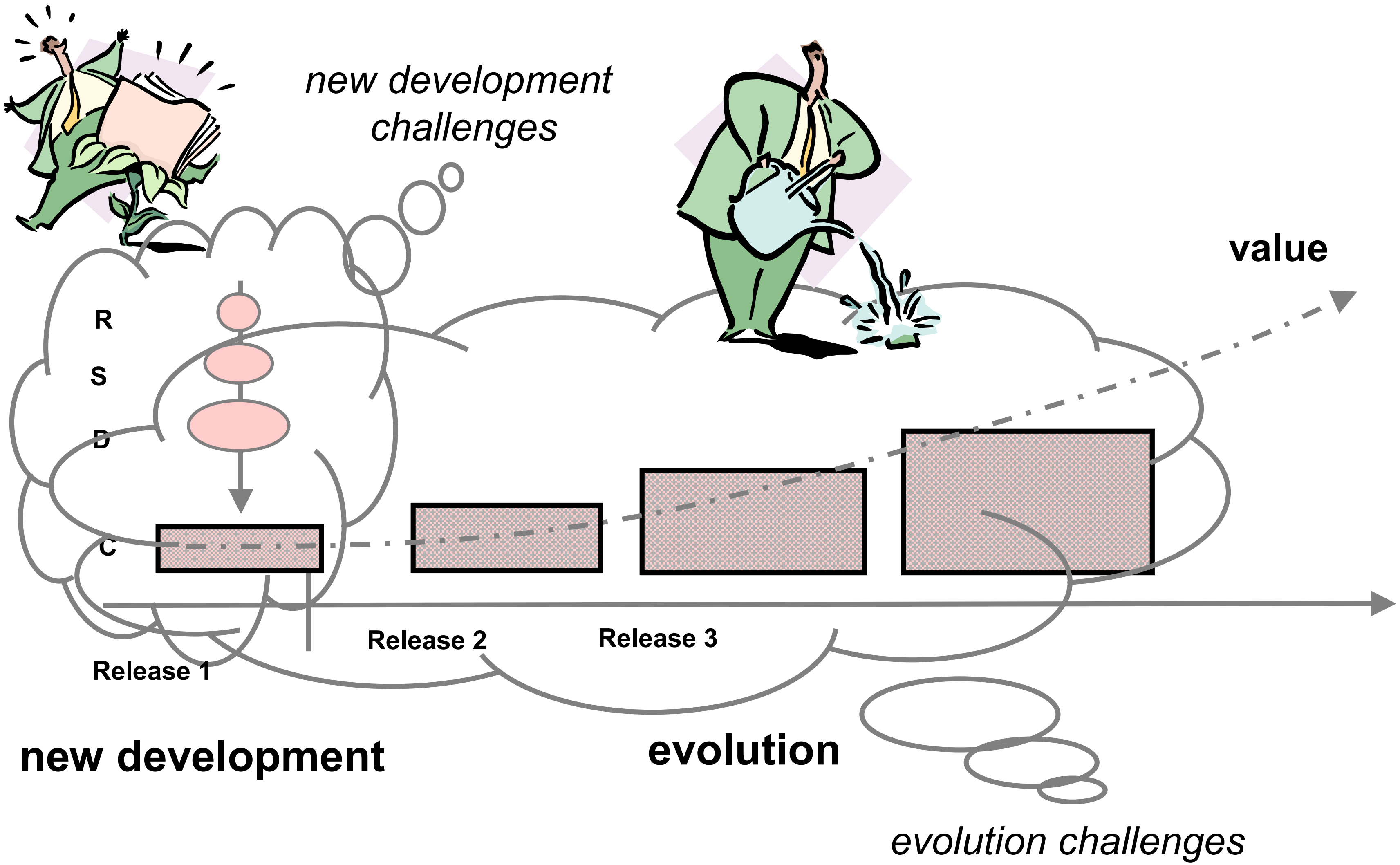


Adapted from Jacobson, et.al., 1997

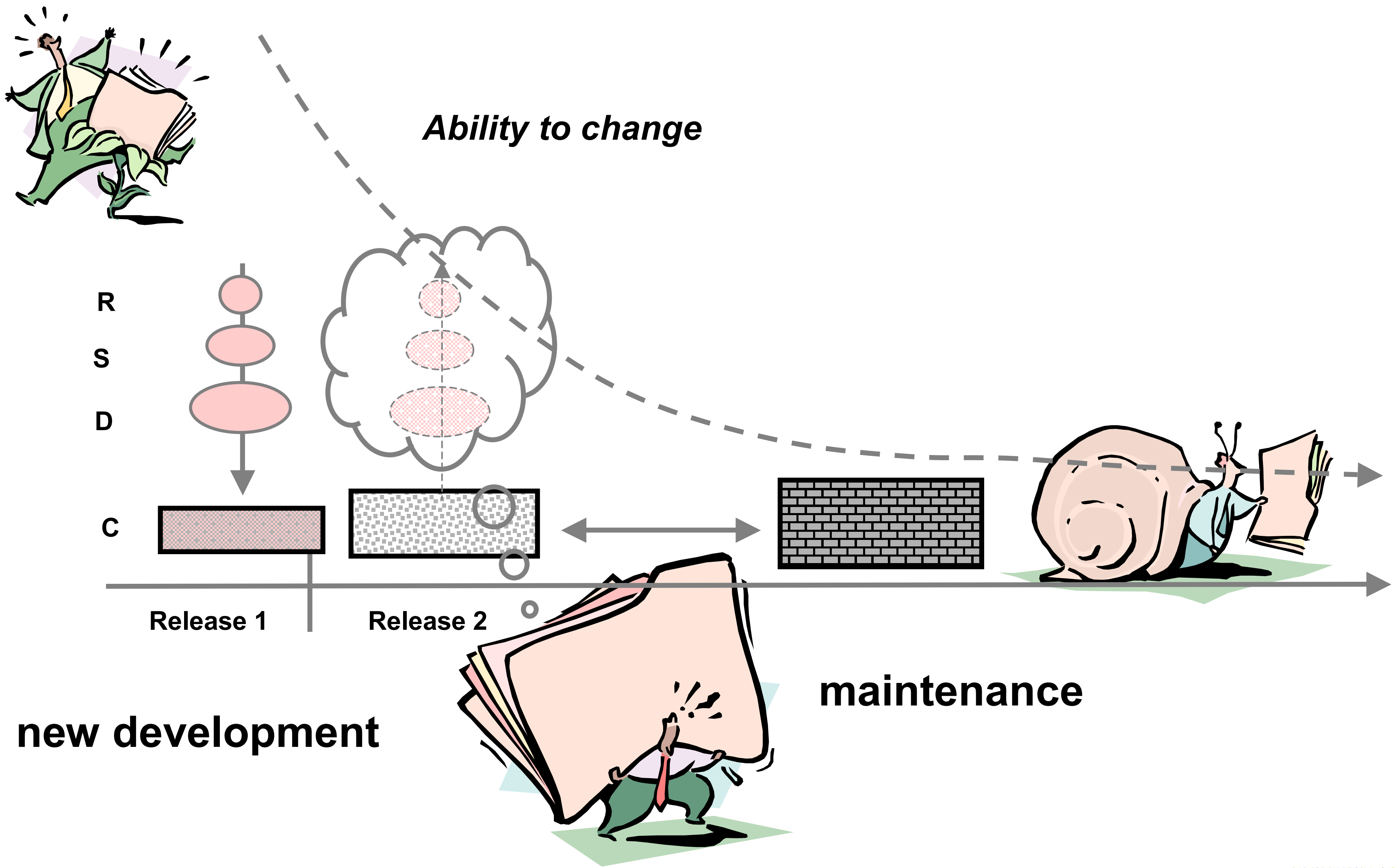
Reuse at different layers



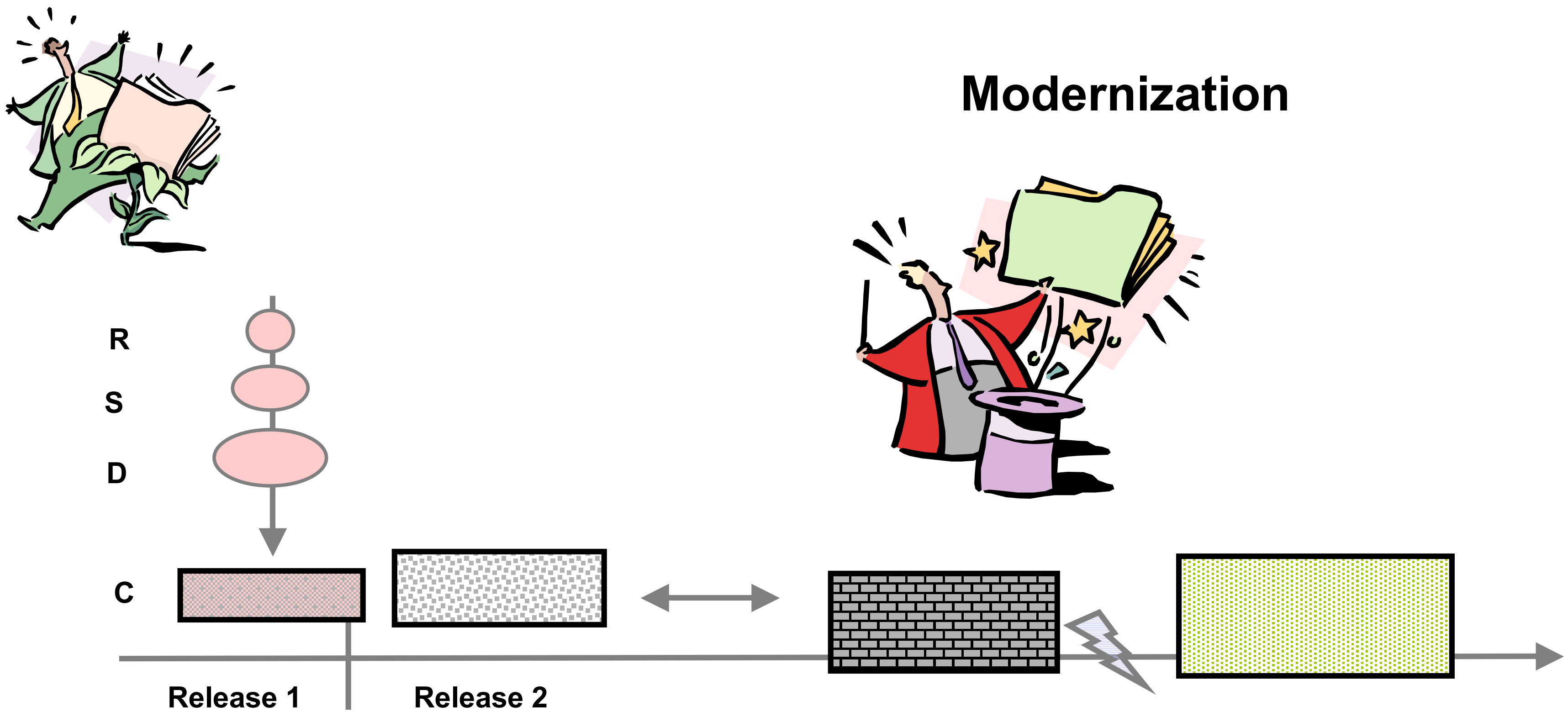
Complete life-cycle: initial release and post-build evolution



Maintenance in a complete life-cycle



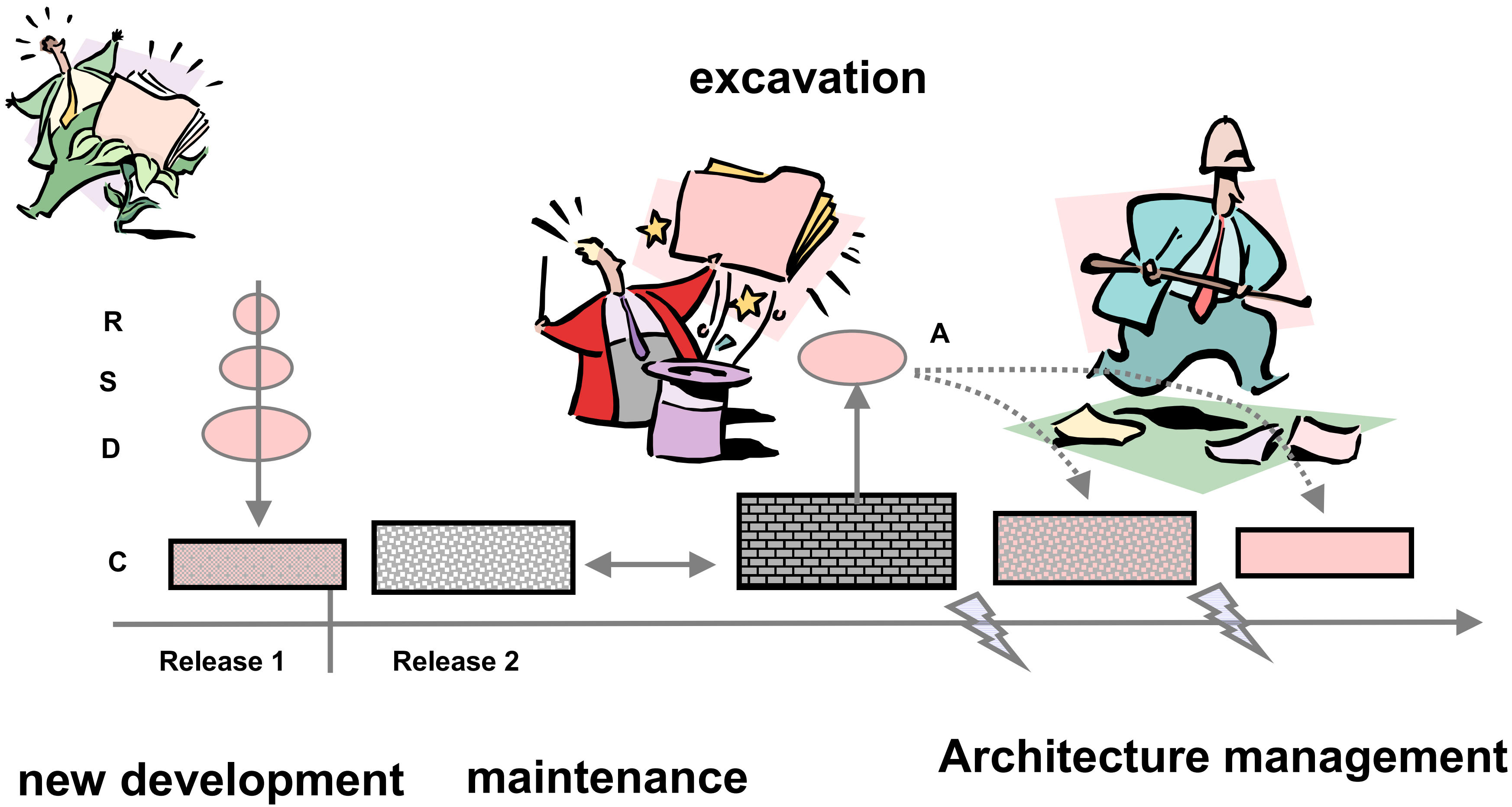
Modernization in a complete life-cycle



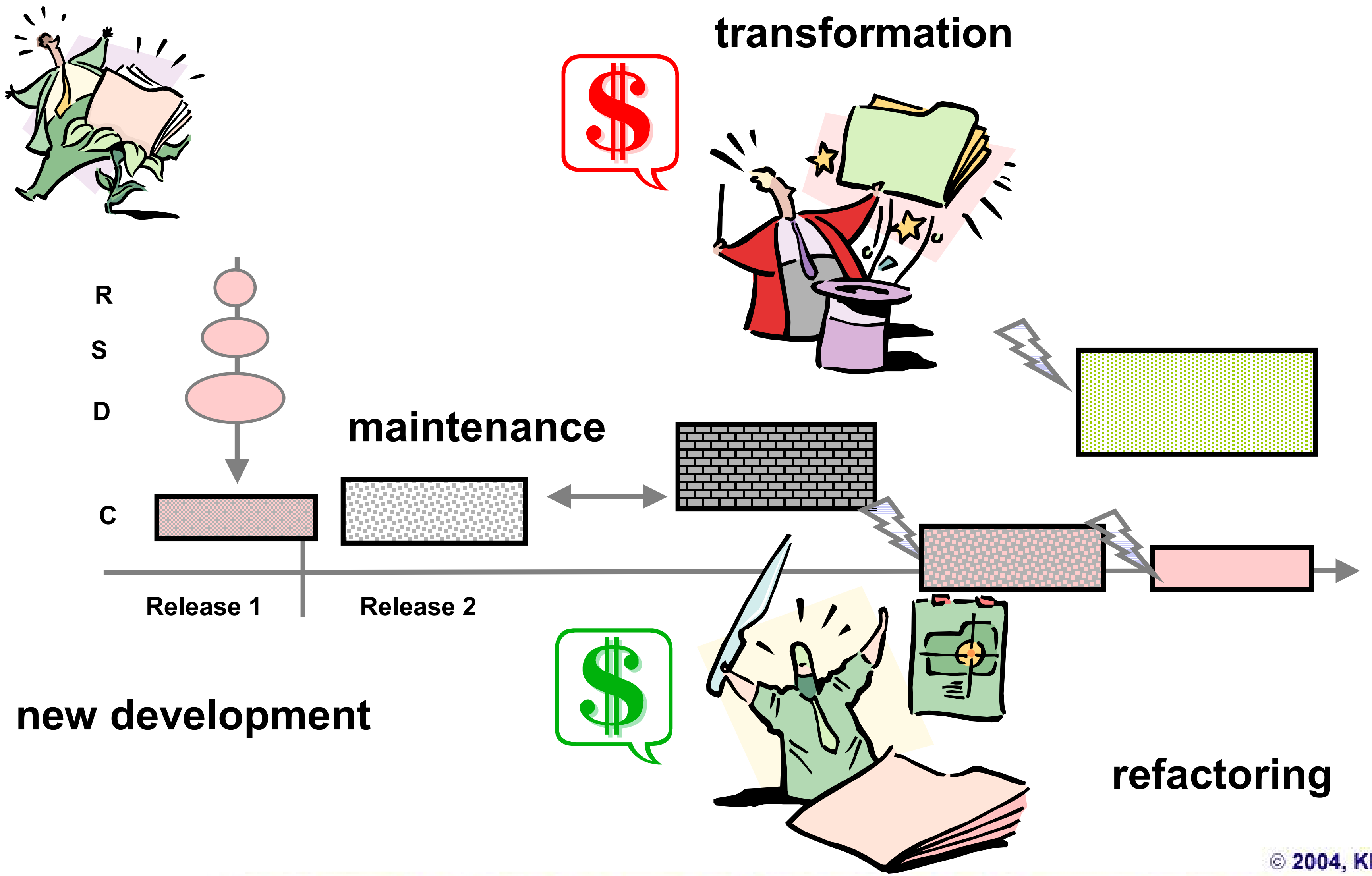
new development **maintenance**

… this includes redevelopment

Managed Architecture in complete life cycle



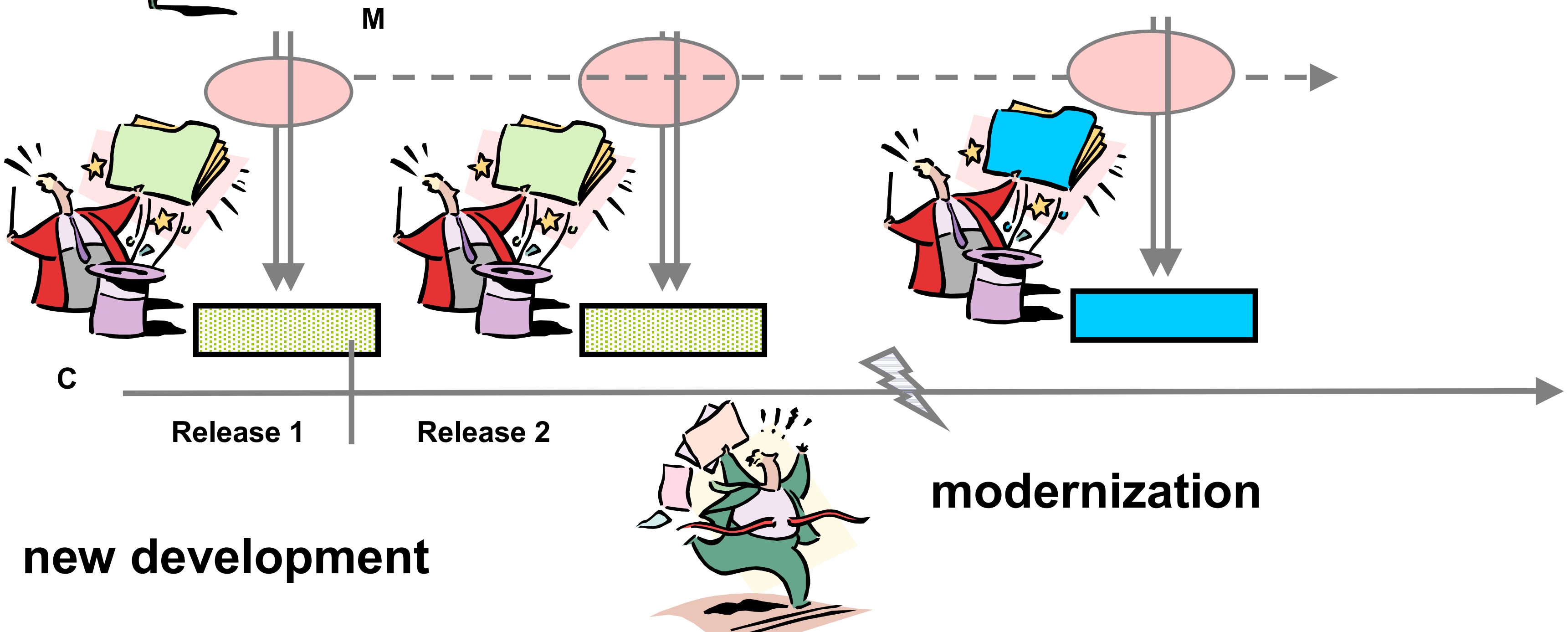
Modernization: Big-bang transformation vs Refactoring



MDA addresses the challenges of a complete life-cycle

…by addressing program comprehension challenge…

…and modernization challenges…

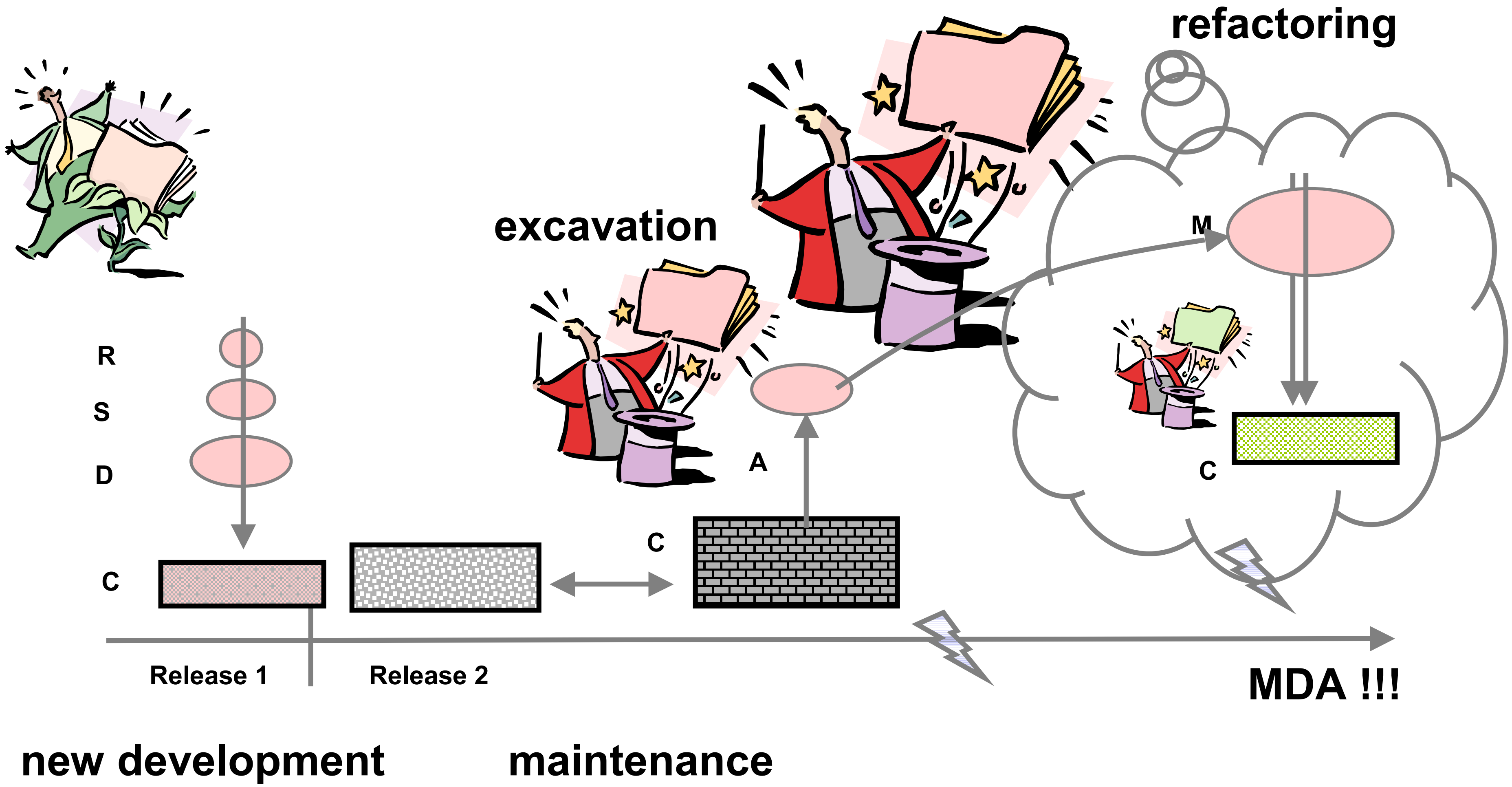


modernization

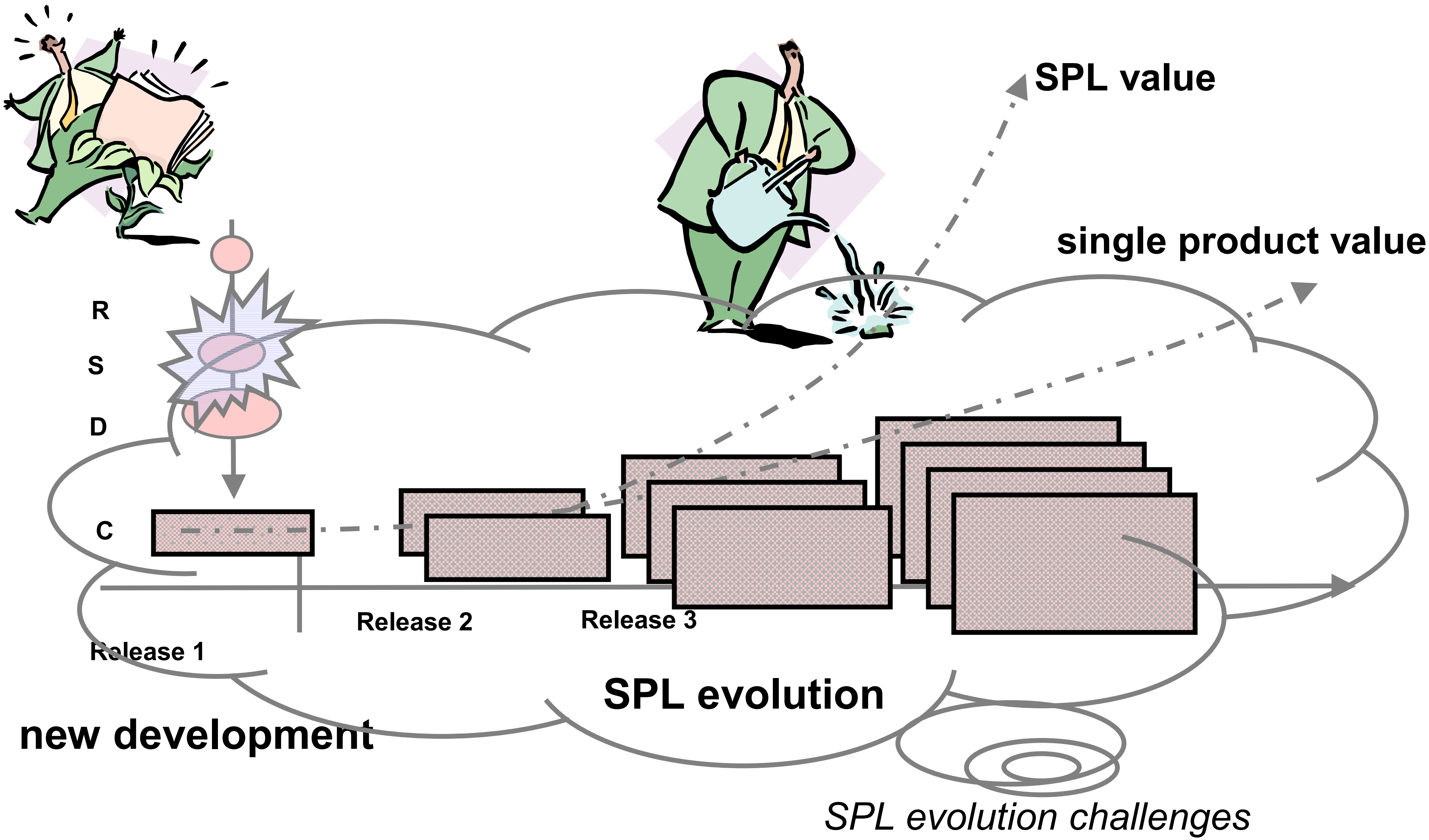
new development

maintenance=continued development

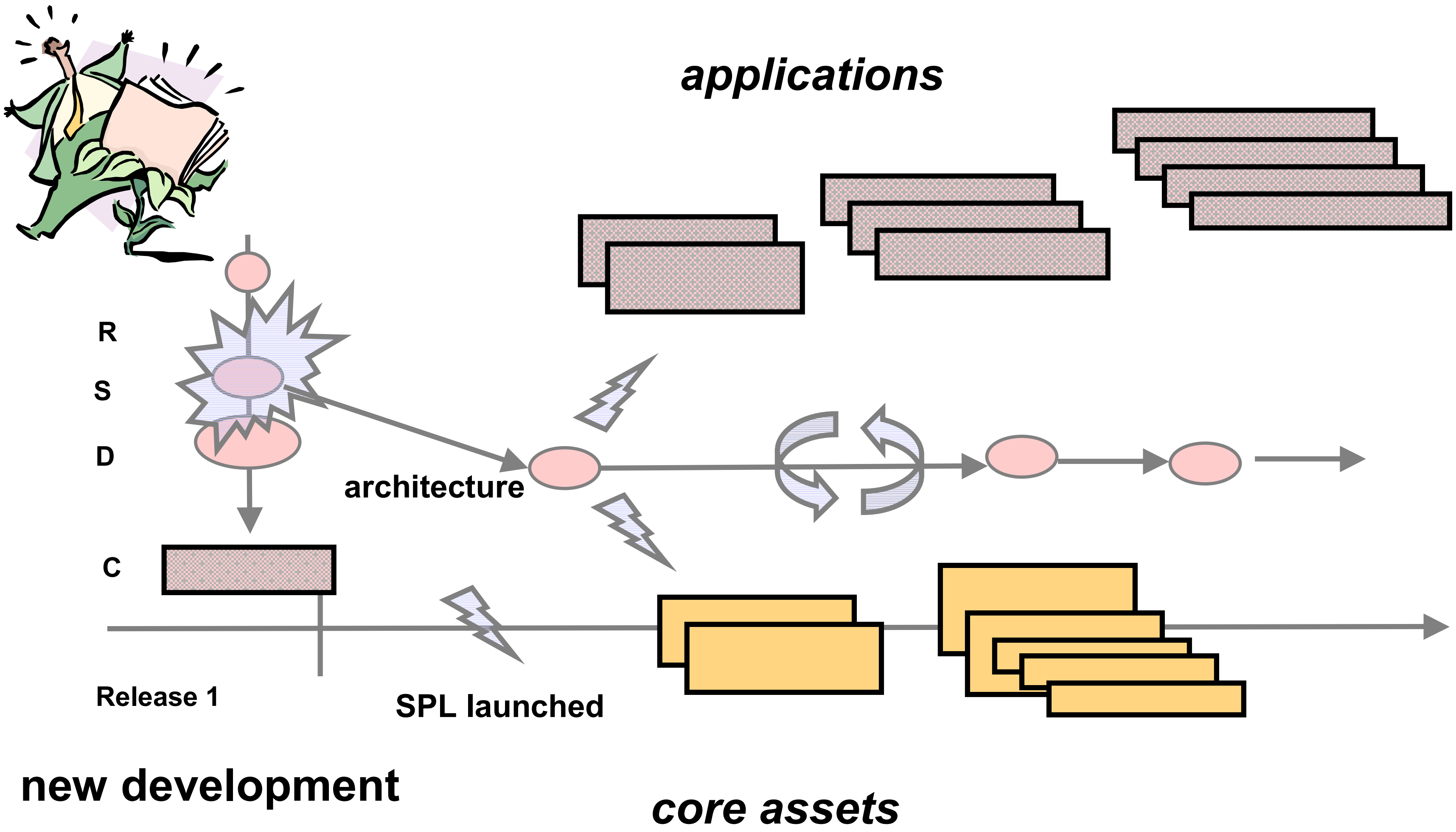
Managed Architecture can kick-start MDA



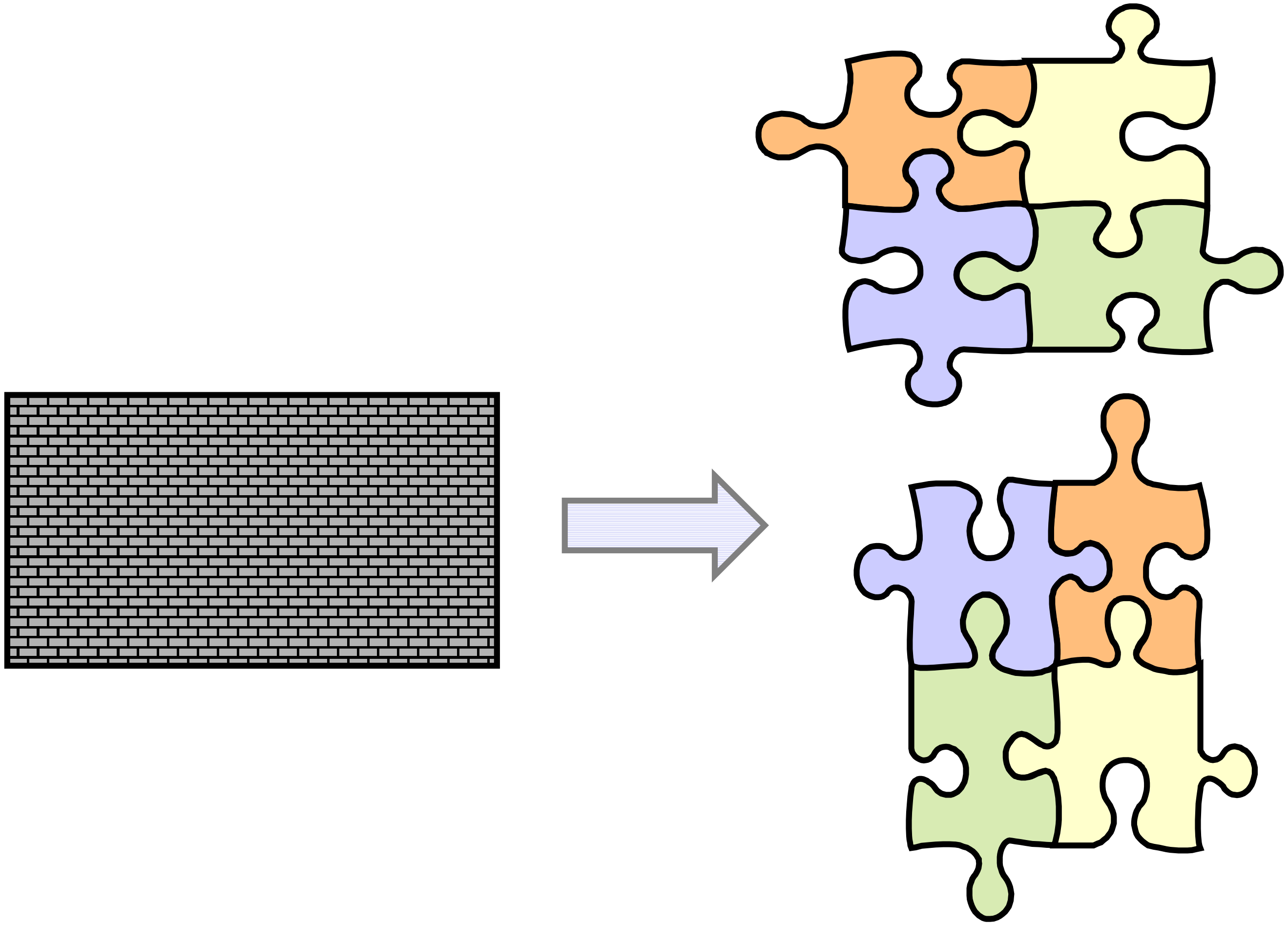
Mega life-cycle: Software Product Lines



Software Product Lines are architecture-centric



SPL require tight architecture management



Architecture Capability Maturity

- **Single aspect of SEI-CMM – refers to the capability of an organization to manage architecture in the complete life-cycle**
- **Level 1:** *Initial architecture*
- **Level 2:** *Repeatable architecture*
  - Some packaging rules, some use of libraries, some code reuse
- **Level 3:** *Defined architecture*
  - Components and their interfaces are formally defined and maintained together with the rest of the code,
  - modelling tools like Rational Rose are used
  - middleware or component environment is used,
- **Level 4:** *Managed architecture*
  - Visualization of existing software is available, feedback between the “as designed” architecture and the “as built” architecture is available, metrics of existing architecture are used, architecture integrity is enforced
- **Level 5:** *Optimizing architecture*
  - On-going architecture improvement is part of the overall development process

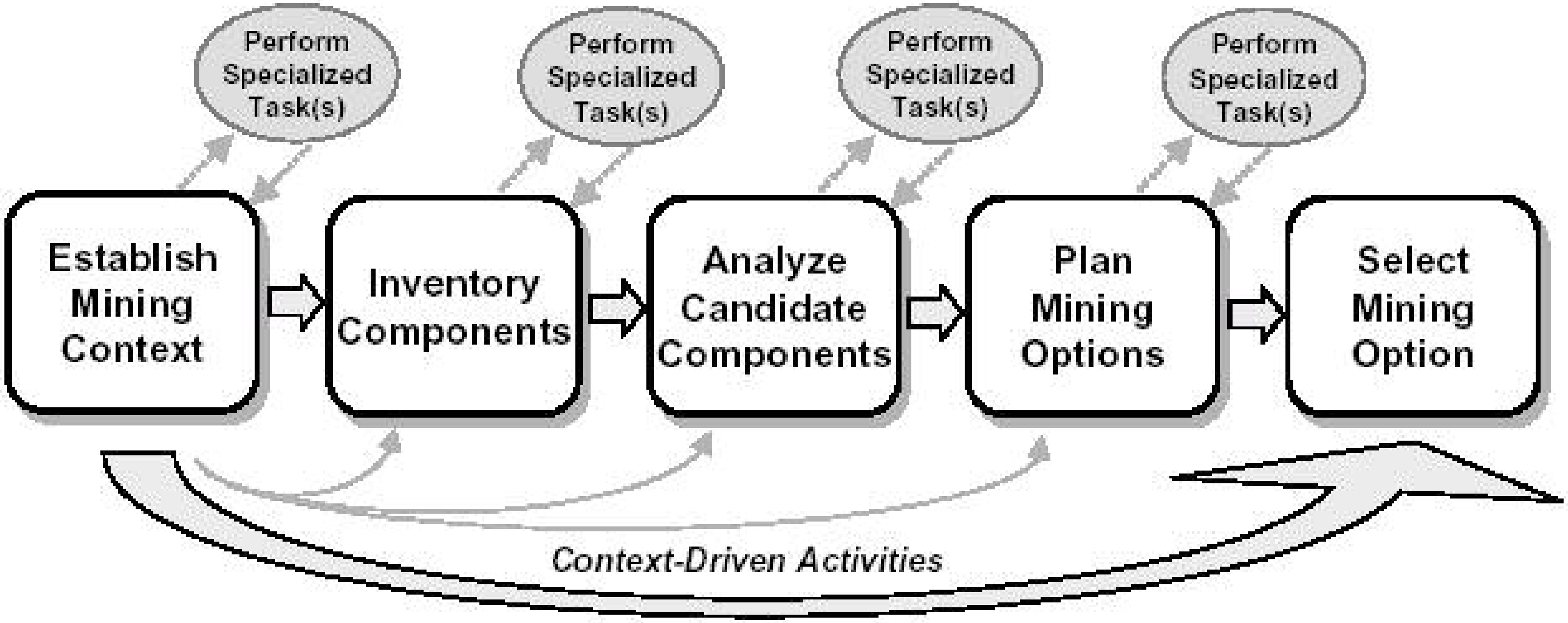
Overview

- **Decision-making process related to reuse**

- **Option Analysis for Reuse**
  - Overview of OAR
  - OAR tasks
  - Example artifacts

- **OAR is a systematic, architecture-centric, decision-making method for mining existing components. OAR is used for modernization or redevelopment of existing system.**
- **OAR is developed in SEI, by Dennis Smith, Liam O'Brien, et.al.**

Overview of OAR



From Smith, et.al., SEI, 1999

OAR: Establish Mining context

- **What:**
  - Interview stakeholders
  - Study existing product line or new system requirements
  - Study existing software assets
  - Understand expectations for mining legacy components
- **This establishes**
  - A baseline set of goals
  - Expectations
  - Component needs
- **This uncovers the program and technical drivers for making decisions**

Adapted from Smith, et.al., SEI, 1999

OAR: Inventory Components

- **What:**
  - Identify product line needs
  - Evaluate legacy components according to screening criteria
  - This results in an inventory of candidate components
- **Tasks:**
  - Identify characteristics of component needs
  - Identify components satisfying criteria
    - Create component table of the legacy components with details of these characteristics
    - Screen components that do not satisfy the required characteristics
  - Match components to product line needs
  - Update the Inventory with more details on candidate components
  - Elicit mining issues and concerns
  - Review OAR schedule

Adapted from Smith, et.al., SEI, 1999

OAR: Analyze Candidate Components

- **What:**
  - Analyze the set of legacy components for the types of changes that are required to mine
- **Tasks:**
  - Select desirable components
    - Determine desirability criteria
    - Screen out components that do not satisfy desirability criteria
  - Identify “As Is” and “Black-Box” (wrapped) components
  - Identify “White-box” components (need to be modified)
  - Determine required changes
    - Types of changes each component needs, cost and effort involved, the level of difficulty and risk, and the comparative cost and effort of developing components from scratch
  - Elicit mining options
  - Review OAR schedule

Adapted from Smith, et.al., SEI, 1999

OAR: Plan Mining Options

- **What:**
  - Develop alternatives to mining based on schedule, cost, effort, risk and resource considerations
  - Screen candidate components one more time
  - Analyze the impact of different component aggregations
- **Tasks:**
  - Select favorable components
    - Determine criteria, such as cost or effort
    - Screen out components that do not satisfy criteria
  - Perform component trade-offs
    - Identify one component or combination per product line need
  - Form component options
    - Develop criteria for aggregation
    - Aggregate components
  - Determine comparative cost and effort
  - Analyze difficulty and risk
  - Elicit mining options
  - Update OAR schedule

Adapted from Smith, et.al., SEI, 1999

OAR: Select Mining Option

- **What:**
  - Select the best mining option or combination of options by balancing program and technical considerations
  - Prepare report
- **Tasks:**
  - Choose best option
    - Determine drivers for selecting among options
    - Select an option or a combination of them
  - Verify option
  - Identify component needs satisfied
    - Complete the final list of component needs satisfied and not satisfied through options selected
  - Present findings
  - Produce summary

Adapted from Smith, et.al., SEI, 1999

Example of Component Table

Part 1 of 3 — **Legacy System Software**

COMPONENT NEEDS	Legacy System Software Components	Program Language	Characteristics Directory Name	Number of Modules	Characteristics Compilable on SGI Machine	Size[1] Lines of Code (LOC)	Complexity
1. Simulation Gateways							
	Sim1 Gateway	C	cag	17	Yes	2,297	Low
	Sim2 Gateway	C	dwg	57	Yes	4,877	Low
2. Message Gateways							
	C-Source	C	csg	45	Yes	4,186	High
3. Truth Model							
	Objects	Fortran	ojo	65	Yes	9,202	Med
	Event	Fortran	tho	39	Yes	4,412	Med
4. ETS & Message Generation							
	Event Track	C	tsd	89	Yes	5,417	Med / High
5. Man Machine Interface						30,391	
	TBD					Total	
6. Executive							
	TBD						

•••

From Smith, et.al., SEI, 1999

Example of Component Table

Part 2 of 3

Legacy System Software

	Characteristics			Characteristics				
Legacy System Software Components	Coupling	Cohesiveness	Black Box / White Box Suitability	Level of Changes	Level of Granularity	Support Software Required	Level of Difficulty[3]	Level of Risk
Sim1 Gateway	Low	High	BB	None	OK	S&D Files[2]	1	Low
Sim2 Gateway	Low	High	BB	None	OK	S&D Files	1	Low
C-Source	Low	High	BB	None	OK	S&D Files	1	Low
Objects	High	High	WB (minor)	Minor (10%)	Adjust	S&D Files	2	High[4]
Event	High	High	WB (minor)	Minor (10%)	Adjust	S&D Files	2	High[4]
Event Track	Low	Low	WB (major)	Major (50%)	Adjust	S&D Files	4	Low[5]
TBD								
TBD								

From Smith, et.al., SEI, 1999

Example of Component Table

Part 3 of 3

Legacy System Software

Software Components	Mining Effort[6] (mm)	Mining Cost[7]	New Development Effort (mm)	New Development Cost[8]	Comparative Cost[7] of Mining	Comparative Effort[6] of Mining
Sim1 Gateway	0.1	$1,000	7.7	$76,567	1%	1%
Sim2 Gateway	0.1	$1,000	16.3	$162,567	1%	1%
C-Source	0.1	$1,000	14.0	$139,533	1%	1%
Objects	2.2	$22,000	30.7	$306,733	7%	7%
Event	1.3	$13,000	14.7	$147,067	9%	9%
Event Track	8	$80,000	18.1	$180,567	44%	44%
TOTALs	12	$118,000	101	$1,013,033	12%	12%
TBD						
TBD						

•••

From Smith, et.al., SEI, 1999

Example of Options Table

Option No.	Legacy System Software Components	Support Software Required	Level of Risk	Level of Difficulty	Mining Effort[1] (mm)	Mining Cost[1]	New Development Effort (mm)	New Development Cost	Comparative Cost of Mining	Comparative Effort of Mining
1	Event Track	S&D Files	Low	4	8	$80,000	18.1	$180,567	44%	44%
	Option Summation		**Low**	**4**	**8**	**$80,000**	**18.1**	**$180,567**	**44%**	**44%**
2	Objects	S&D Files	High	2	2.2	$22,000	30.7	$306,733	7%	7%
	Event	S&D Files	High	2	1.3	$13,000	14.7	$147,067	9%	9%
	Option Summation		**High**	**2**	**3.5**	**$35,000**	**45.4**	**$453,800**	**8%**	**8%**
3	Sim1 Gateway	S&D Files	Low	1	0.1	$1,000	7.7	$76,567	1%	1%
	Sim2 Gateway	S&D Files	Low	1	0.1	$1,000	16.3	$162,567	1%	1%
	C-Source	S&D Files	Low	1	0.1	$1,000	14	$139,533	1%	1%
	Option Summation		**Low**	**1**	**0.3**	**$3,000**	**38**	**$378,667**	**1%**	**1%**

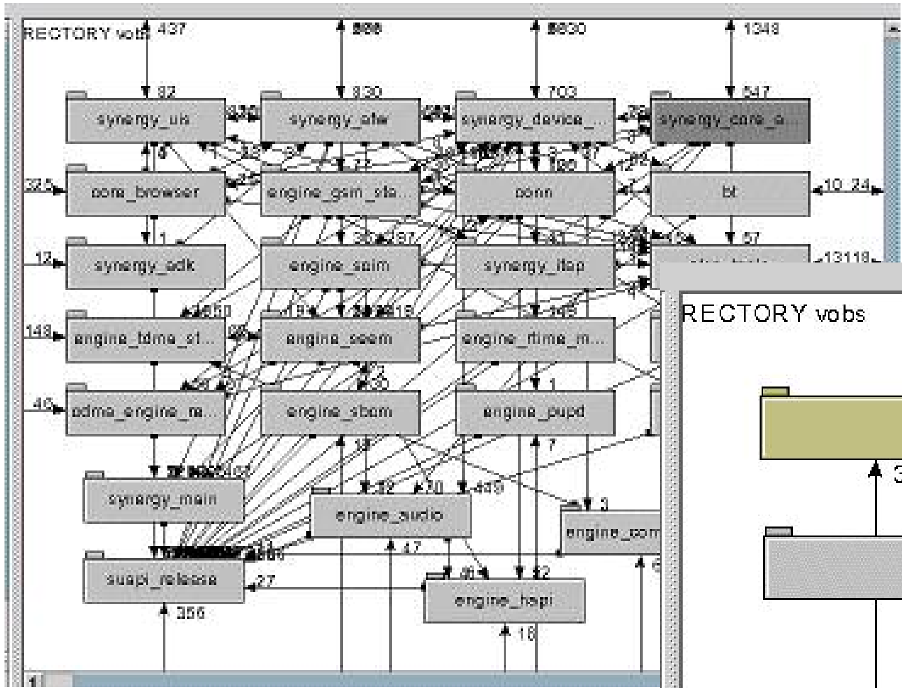
[1] Note: Mining Effort and Cost do not include effort and cost to convert scripts and data files that are part of the support software

From Smith, et.al., SEI, 1999

Overview

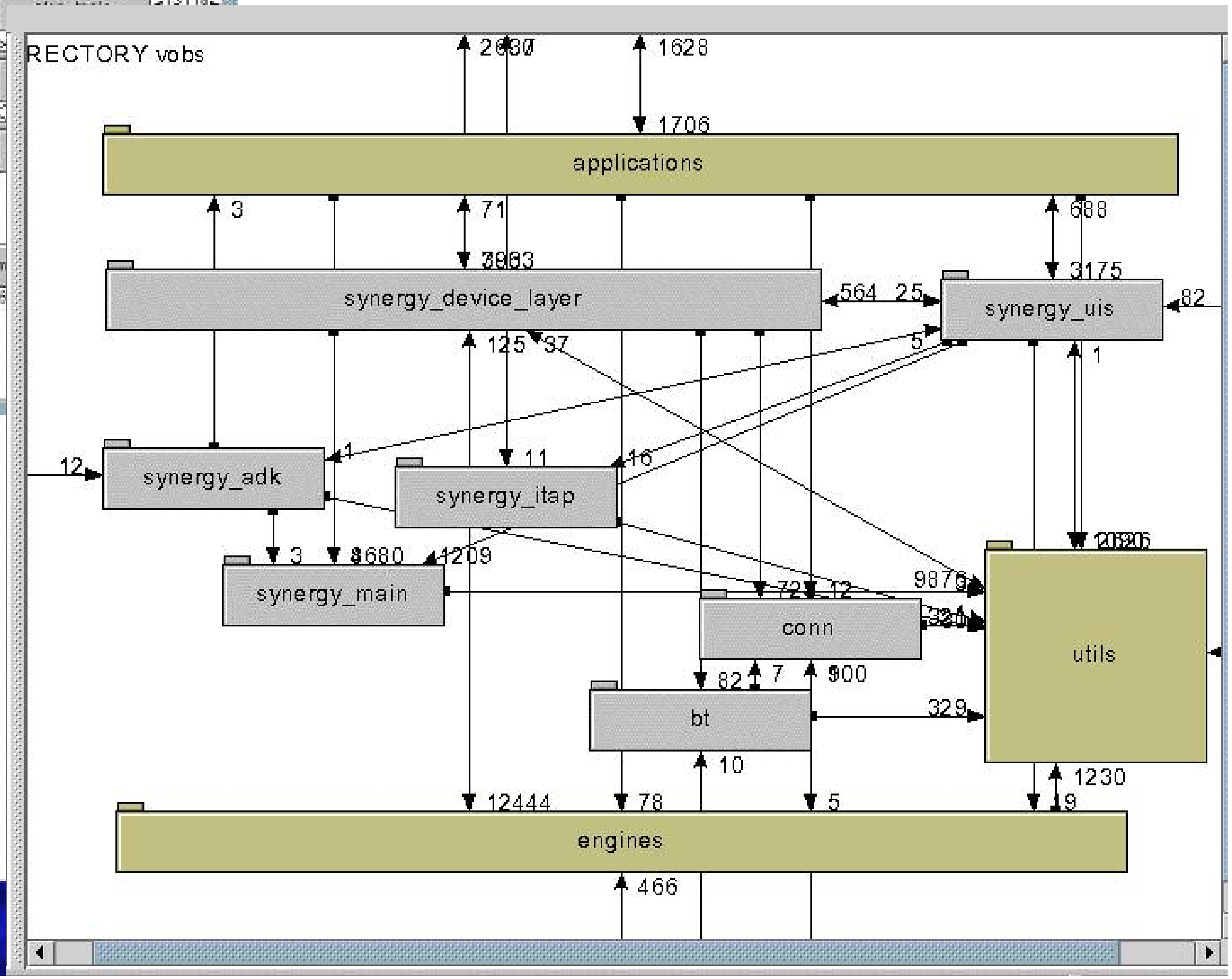
- **Architecture Excavation and Refactoring**

Excavated high-level architecture



before

after



Klocwork

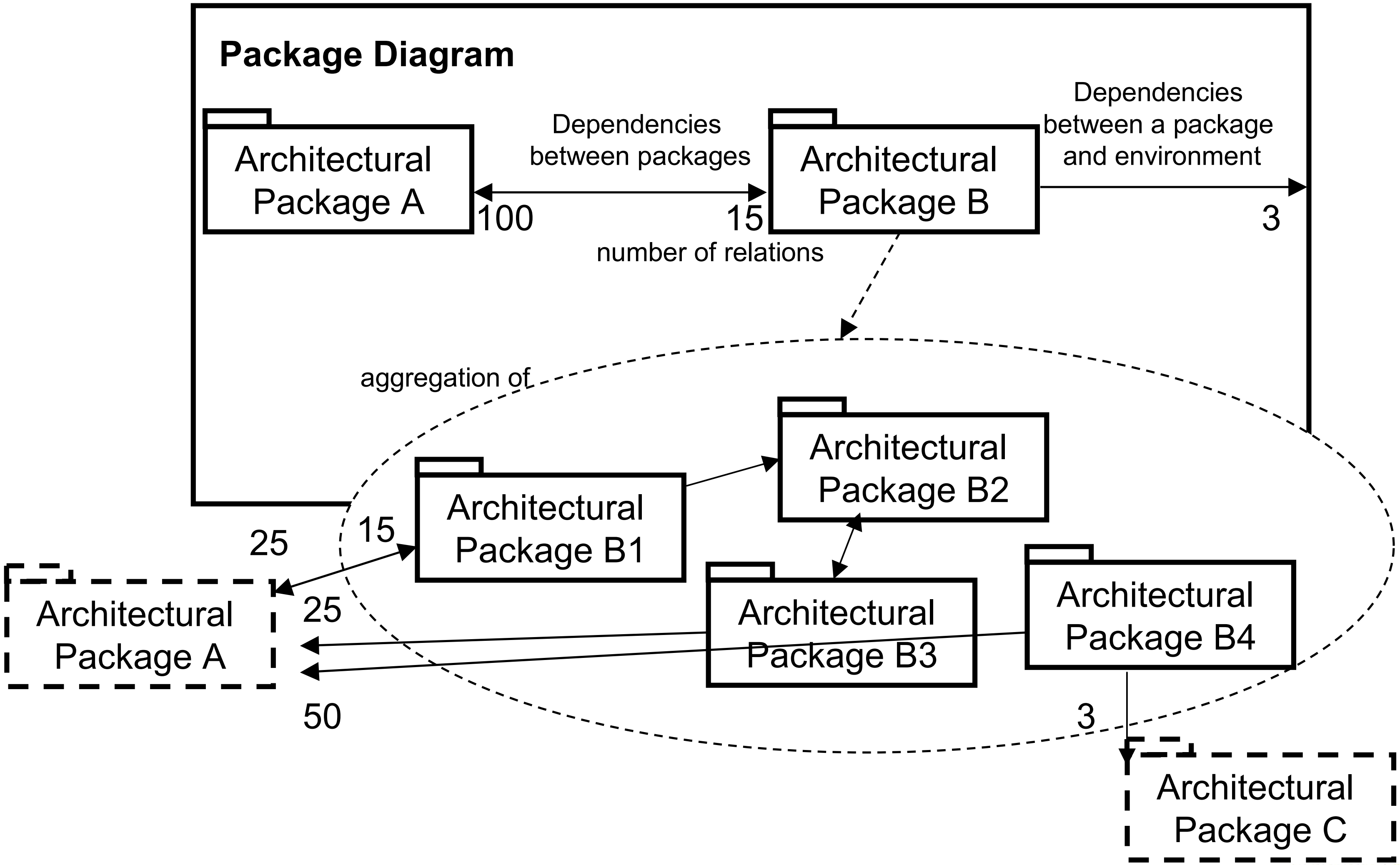
Klocwork Architecture Excavation Methodology

- **Focus:**
  - Containers, interfaces, dependencies on top of entities and relationships
- **Container Models**
  - Are scalable and precise; can be used for both abstraction and refactoring
  - Not UML, because of scalability, the need to evolve with the code, and specific “existing code” understanding concerns, like links to the code, navigation, etc.
  - Transition to UML is straightforward
- **Strategy**
  - Top-down
  - As deep as necessary, as shallow as possible
  - incremental
- **Operations:**
  - Aggregation of entities into bigger containers
  - Refactoring (moving entities between containers)

Container models

- **Represent "containers" and relations between "containers":**
  - each "container" has dependencies on other "containers"
  - each "container" provides an API to other "containers"
- **This model is scalable:**
  - aggregation of "containers" is another "container"
  - The aggregation depends on everything that individual parts depended on
  - The aggregation provides the union of APIs, provided by individual parts
- **Model can be refactored**
  - subcontainers can be moved from one container to another, model shows how dependencies and APIs change
- **This model is precise**
  - With respect to the contents of aggregations
  - With respect to APIs
- **This model is meaningful and useful**
  - Leaf "containers" can be procedures, variables, files, etc.
  - Leaf relations (APIs) are e.g. procedure-calls-procedure, etc.
- **Model can be preserved and automatically updated as changes are made to software**
  - Leaf containers and their relations are automatically extracted from source; the model stores only the hierarchy of containers; relations are recalculated on-the-fly

Container models

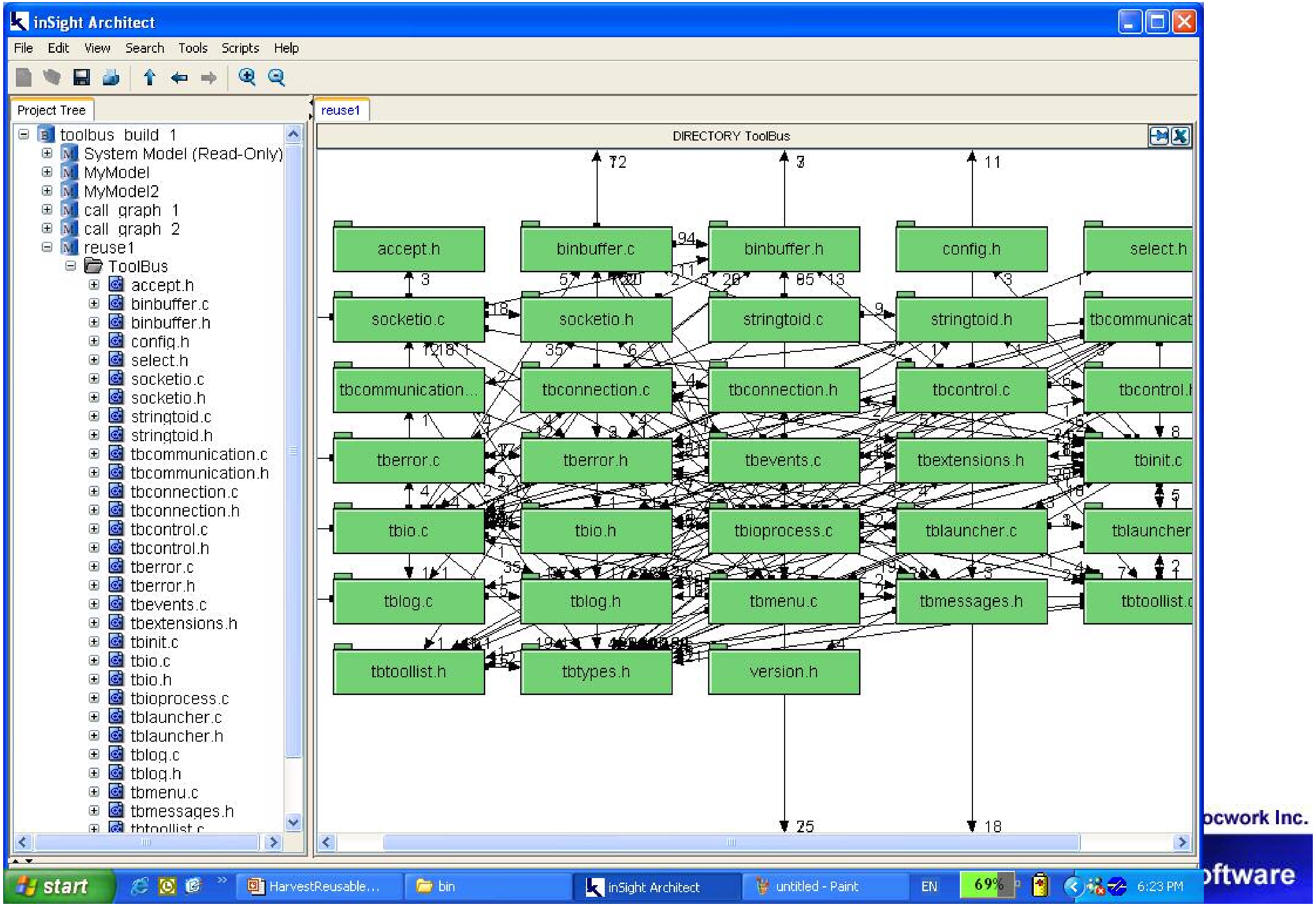


Overview

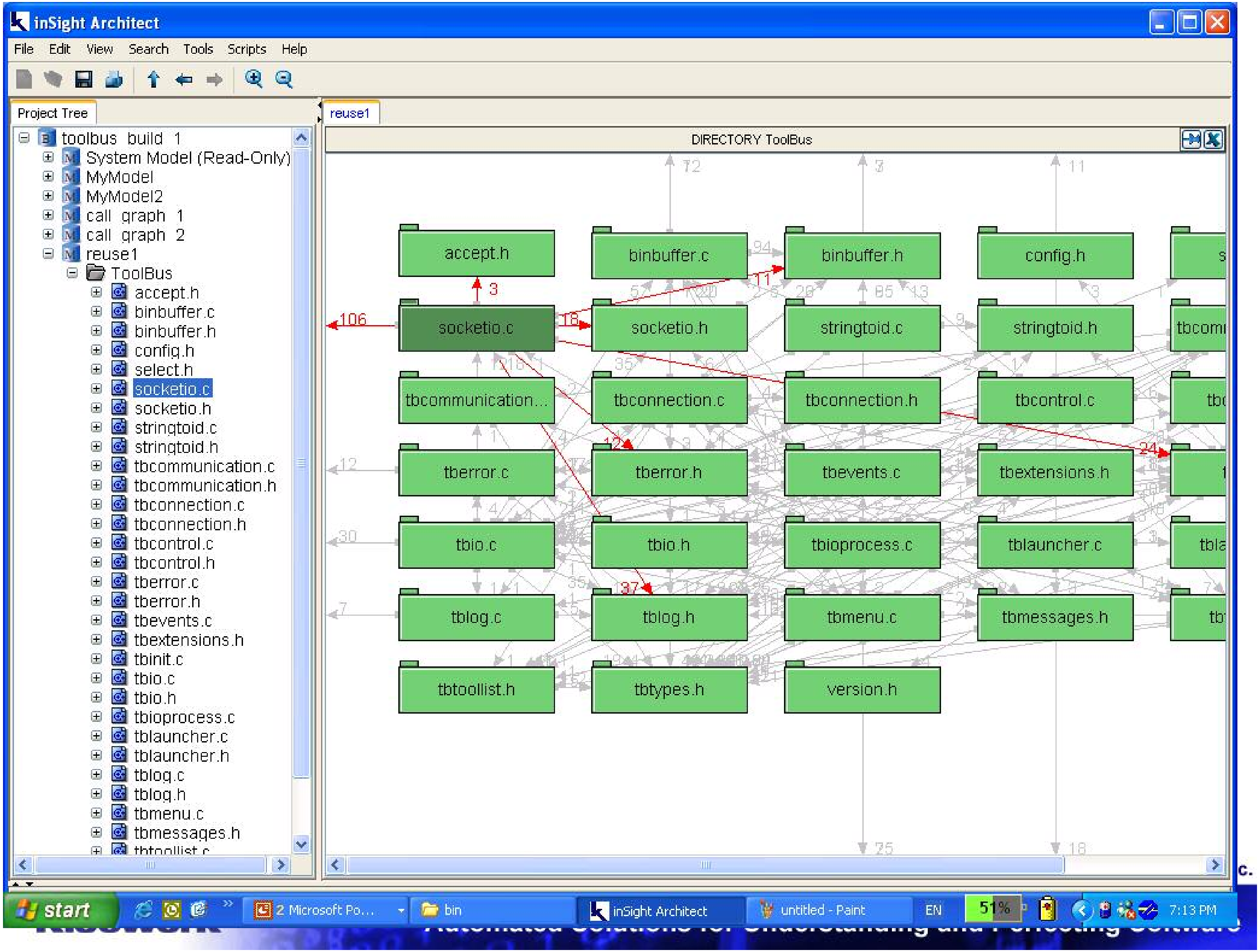
- **Harvesting as architectural refactoring**
  - First, you need to excavate your architecture, as a precise model
  - Reuse requires the following steps:
    - Identify/localize/collect
    - Isolate component
    - Introduce a boundary
    - Implement refactoring
  - Particular challenges
    - Challenges of component identification (need to understand)
    - Challenges of component isolation
    - Challenges of soft boundaries
    - Reusable components and architecture erosion
    - Reusable components and cyclic dependencies

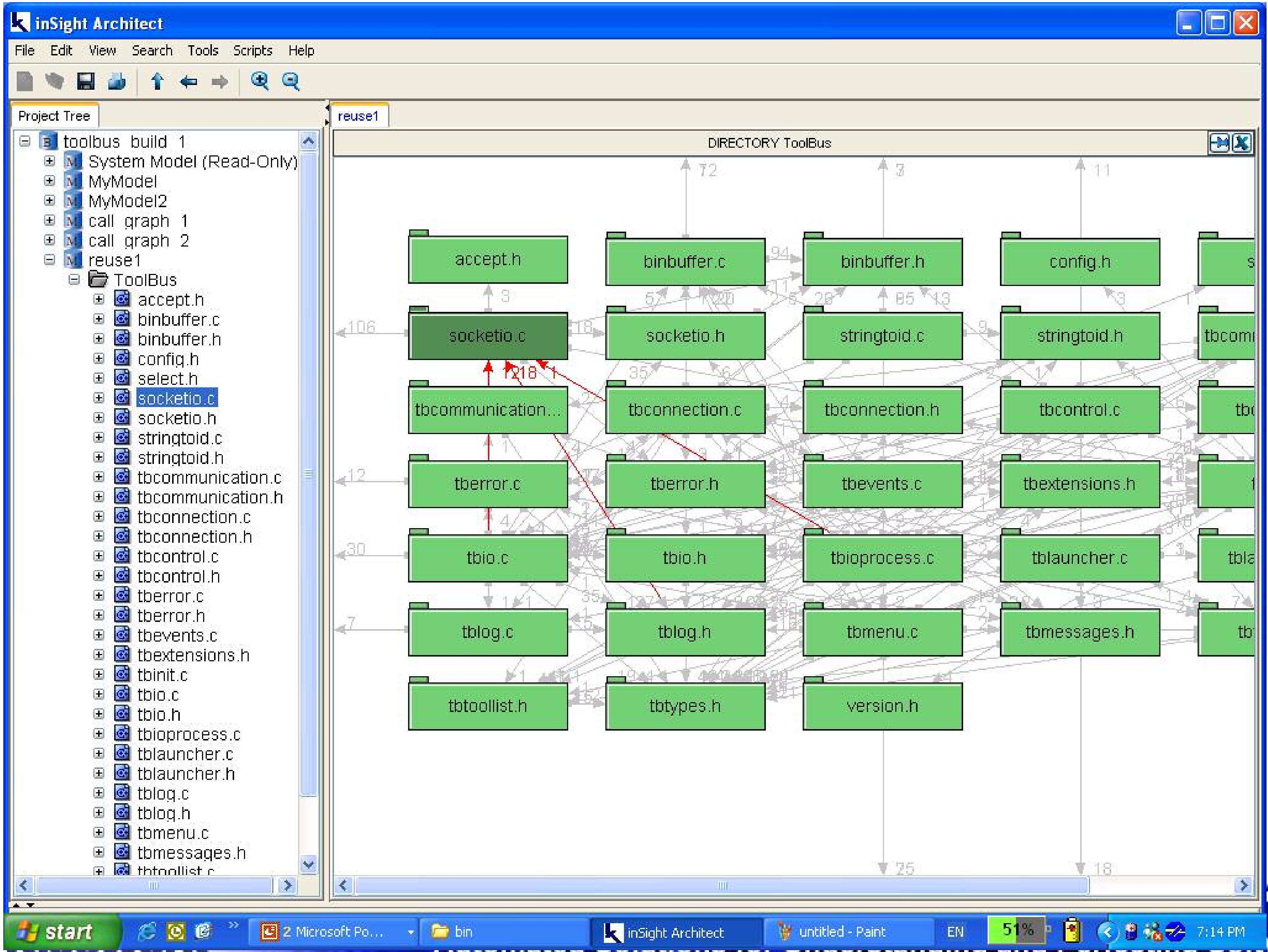
Overview

- **Examples of Isolating Components via Refactoring**



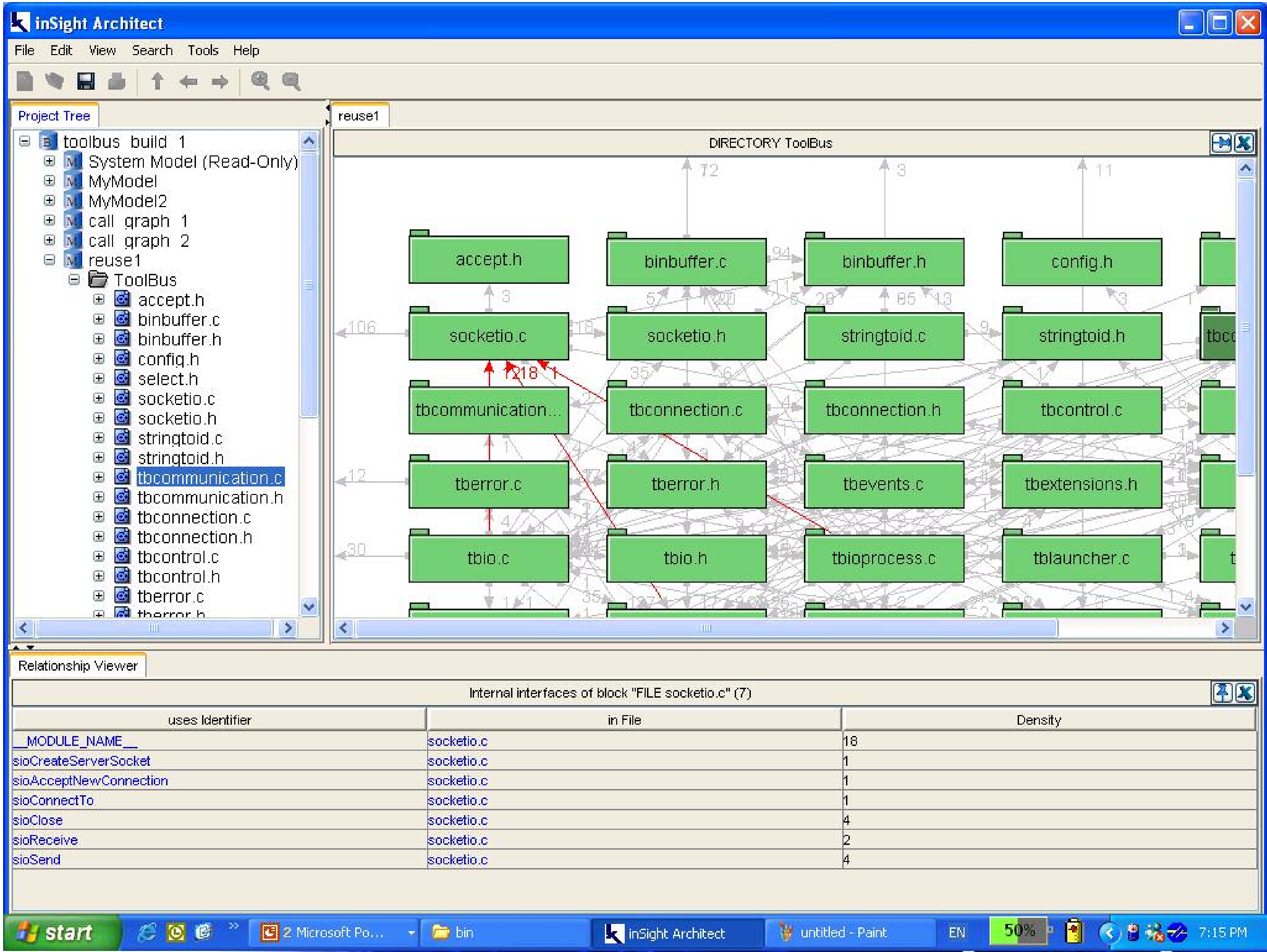
inSight Architect
File Edit View Search Tools Scripts Help
Project Tree
toolbus build 1
System Model (Read-Only)
MyModel
MyModel2
call graph 1
call graph 2
reuse1
ToolBus
accept.h
binbuffer.c
binbuffer.h
config.h
select.h
socketio.c
socketio.h
stringtoid.c
stringtoid.h
tbcommunication.c
tbcommunication.h
tbconnection.c
tbconnection.h
tbcontrol.c
tbcontrol.h
tberror.c
tberror.h
tbevents.c
tbextensions.h
tbinit.c
tbio.c
tbio.h
tbioprocess.c
tblauncher.c
tblauncher.h
tblog.c
tblog.h
tbmenu.c
tbmessages.h
reuse1
DIRECTORY ToolBus
accept.h
binbuffer.c
binbuffer.h
config.h
select.h
socketio.c
socketio.h
stringtoid.c
stringtoid.h
tbcommunicat
tbcommunication...
tbconnection.c
tbconnection.h
tbcontrol.c
tbcontrol.h
tberror.c
tberror.h
tbevents.c
tbextensions.h
tbinit.c
tbio.c
tbio.h
tbioprocess.c
tblauncher.c
tblauncher
tblog.c
tblog.h
tbmenu.c
tbmessages.h
tbtoollist.c
tbtoollist.h
tbtypes.h
version.h
start
HarvestReusable...
bin
inSight Architect
untitled - Paint
EN
69%
6:23 PM
ocwork Inc.
oftware


inSight Architect
File Edit View Search Tools Scripts Help
Project Tree
toolbus build 1
System Model (Read-Only)
MyModel
MyModel2
call graph 1
call graph 2
reuse1
ToolBus
accept.h
binbuffer.c
binbuffer.h
config.h
select.h
socketio.c
socketio.h
stringtoid.c
stringtoid.h
tbcommunication.c
tbcommunication.h
tbconnection.c
tbconnection.h
tbcontrol.c
tbcontrol.h
tberror.c
tberror.h
tbevents.c
tbextensions.h
tbinit.c
tbio.c
tbio.h
tbioprocess.c
tblauncher.c
tblauncher.h
tblog.c
tblog.h
tbmenu.c
tbmessages.h
reuse1
DIRECTORY ToolBus
accept.h
binbuffer.c
binbuffer.h
config.h
socketio.c
socketio.h
stringtoid.c
stringtoid.h
tbcommunication...
tbconnection.c
tbconnection.h
tbcontrol.c
tberror.c
tberror.h
tbevents.c
tbextensions.h
tbio.c
tbio.h
tbioprocess.c
tblauncher.c
tblog.c
tblog.h
tbmenu.c
tbmessages.h
tbtoollist.h
tbtypes.h
version.h
start
2 Microsoft Po...
bin
inSight Architect
untitled - Paint
EN
7:13 PM

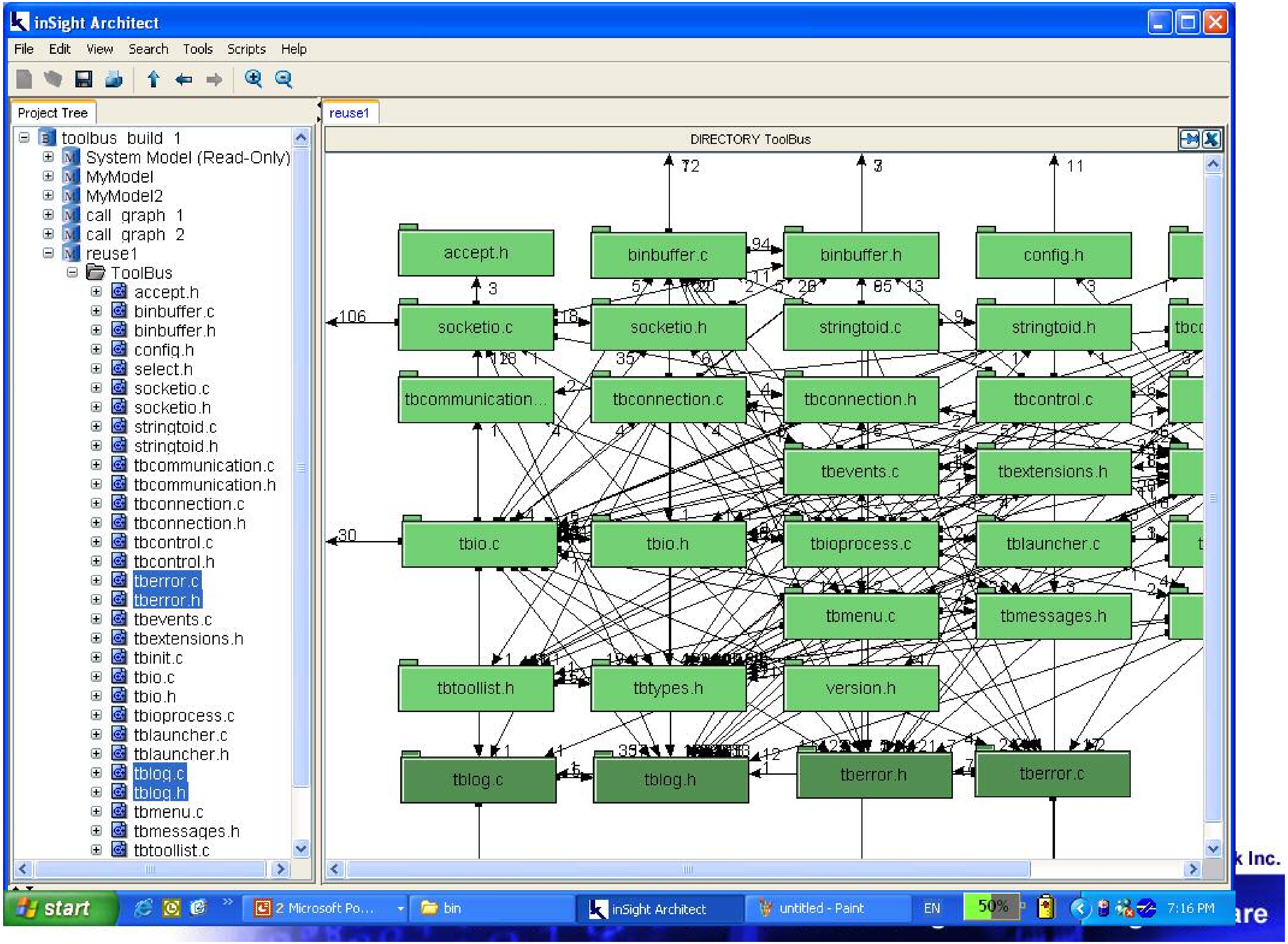


inSight Architect
File Edit View Search Tools Scripts Help
Project Tree
toolbus build 1
System Model (Read-Only)
MyModel
MyModel2
call graph 1
call graph 2
reuse1
ToolBus
accept.h
binbuffer.c
binbuffer.h
config.h
select.h
socketio.c
socketio.h
stringtoid.c
stringtoid.h
tbcommunication.c
tbcommunication.h
tbconnection.c
tbconnection.h
tbcontrol.c
tbcontrol.h
tberror.c
tberror.h
tbevents.c
tbextensions.h
tbinit.c
tbio.c
tbio.h
tbioprocess.c
tblauncher.c
tblauncher.h
tblog.c
tblog.h
tbmenu.c
tbmessages.h
reuse1
DIRECTORY ToolBus
accept.h
binbuffer.c
binbuffer.h
config.h
socketio.c
socketio.h
stringtoid.c
stringtoid.h
tbcommunication...
tbconnection.c
tbconnection.h
tbcontrol.c
tberror.c
tberror.h
tbevents.c
tbextensions.h
tbio.c
tbio.h
tbioprocess.c
tblauncher.c
tblog.c
tblog.h
tbmenu.c
tbmessages.h
tbtoollist.h
tbtypes.h
version.h
start
2 Microsoft Po...
bin
inSight Architect
untitled - Paint
EN
7:14 PM

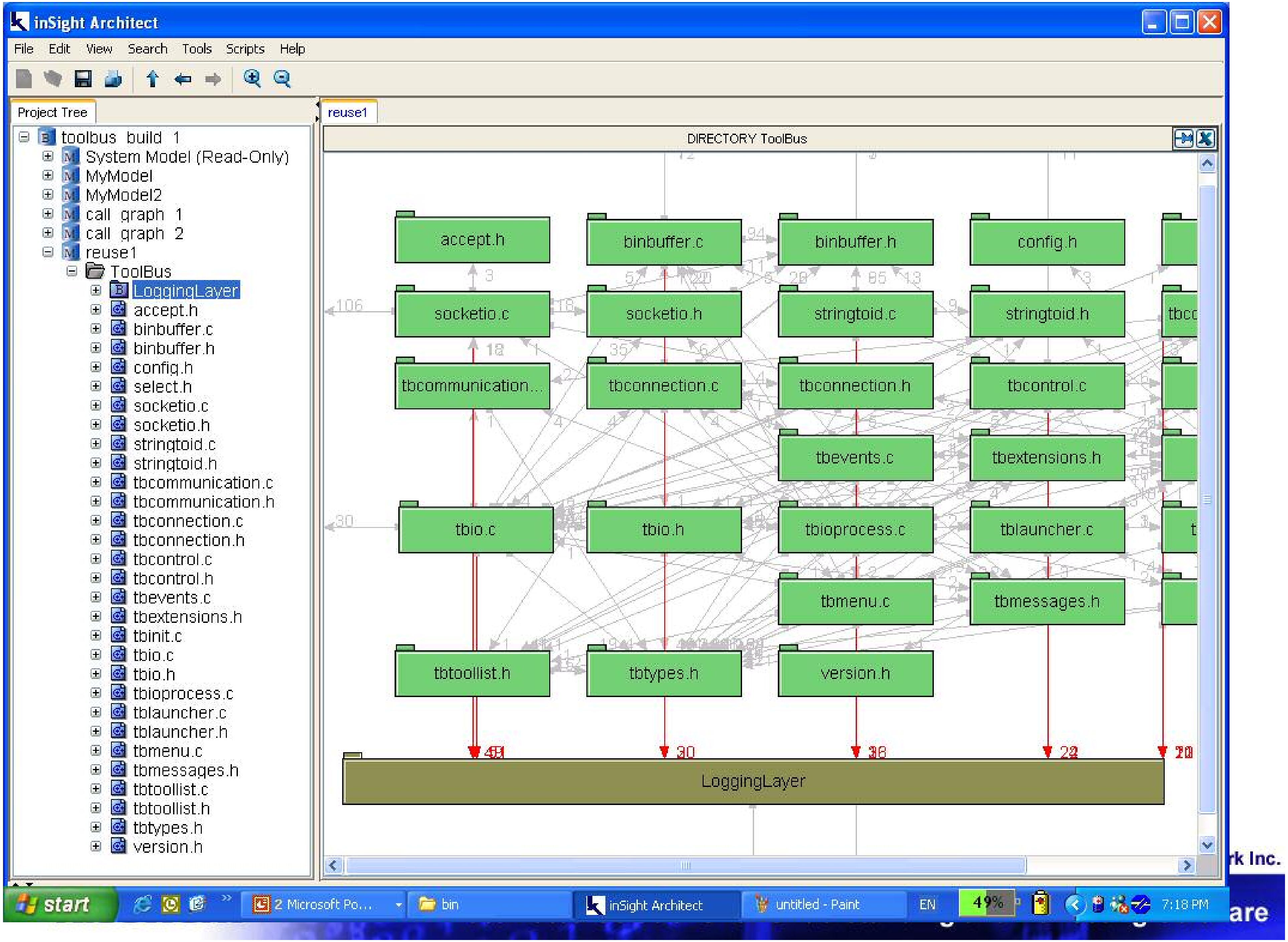
ork Inc.
ware


inSight Architect
File Edit View Search Tools Help
Project Tree
toolbus build 1
System Model (Read-Only)
MyModel
MyModel2
call graph 1
call graph 2
reuse1
ToolBus
accept.h
binbuffer.c
binbuffer.h
config.h
select.h
socketio.c
socketio.h
stringtoid.c
stringtoid.h
tbcommunication.c
tbcommunication.h
tbconnection.c
tbconnection.h
tbcontrol.c
tbcontrol.h
tberror.c
reuse1
DIRECTORY ToolBus
accept.h
binbuffer.c
binbuffer.h
config.h
socketio.c
socketio.h
stringtoid.c
stringtoid.h
tbcommunication...
tbconnection.c
tbconnection.h
tbcontrol.c
tberror.c
tberror.h
tbevents.c
tbextensions.h
tbio.c
tbio.h
tbioprocess.c
tblauncher.c
Relationship Viewer
Internal interfaces of block "FILE socketio.c" (7)
uses Identifier
in File
Density
__MODULE_NAME__ socketio.c 18
sioCreateServerSocket socketio.c 1
sioAcceptNewConnection socketio.c 1
sioConnectTo socketio.c 1
sioClose socketio.c 4
sioReceive socketio.c 2
sioSend socketio.c 4
start
2 Microsoft Po...
bin
inSight Architect
untitled - Paint
EN
50%
7:15 PM
k Inc.

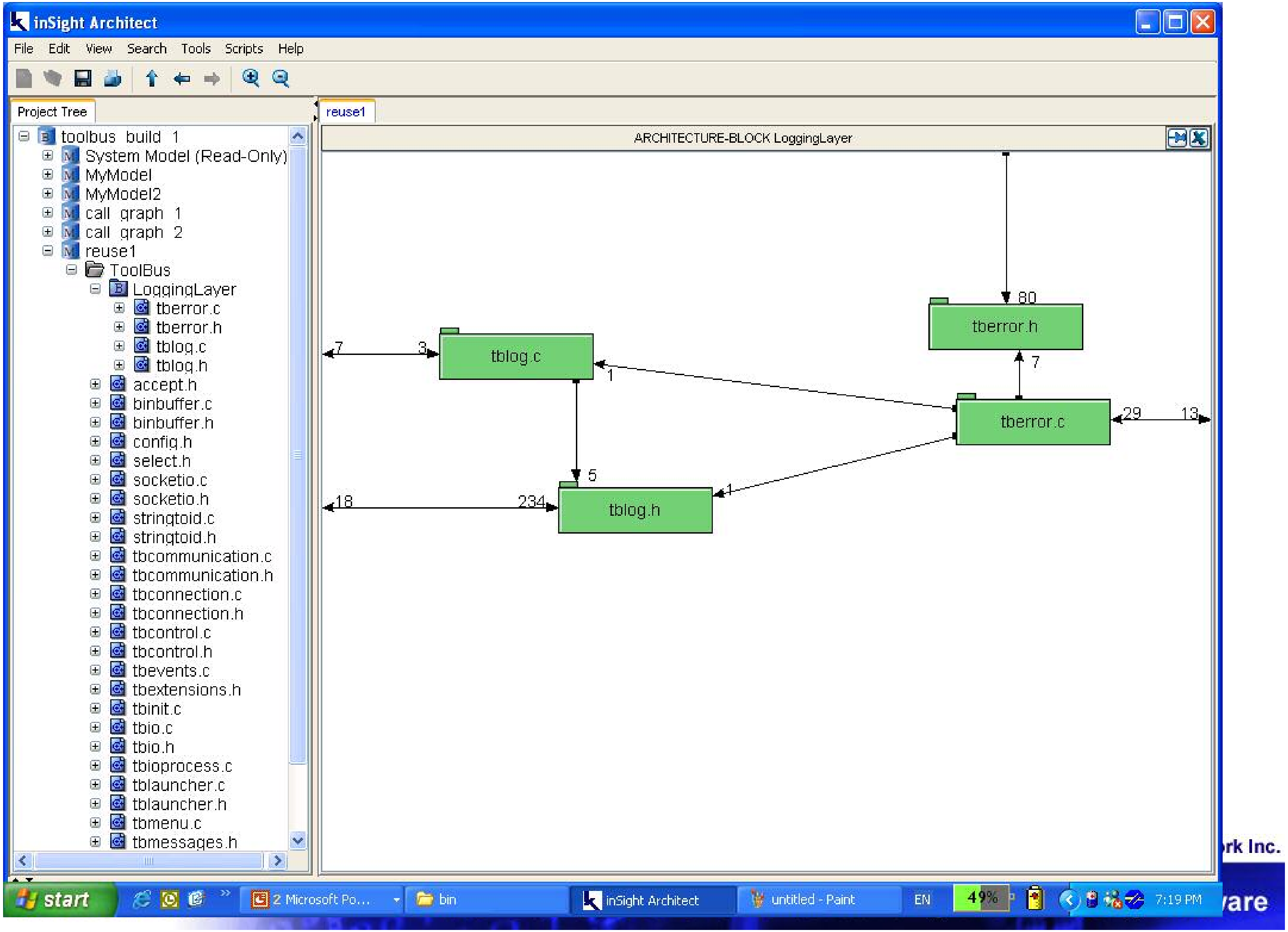
ire

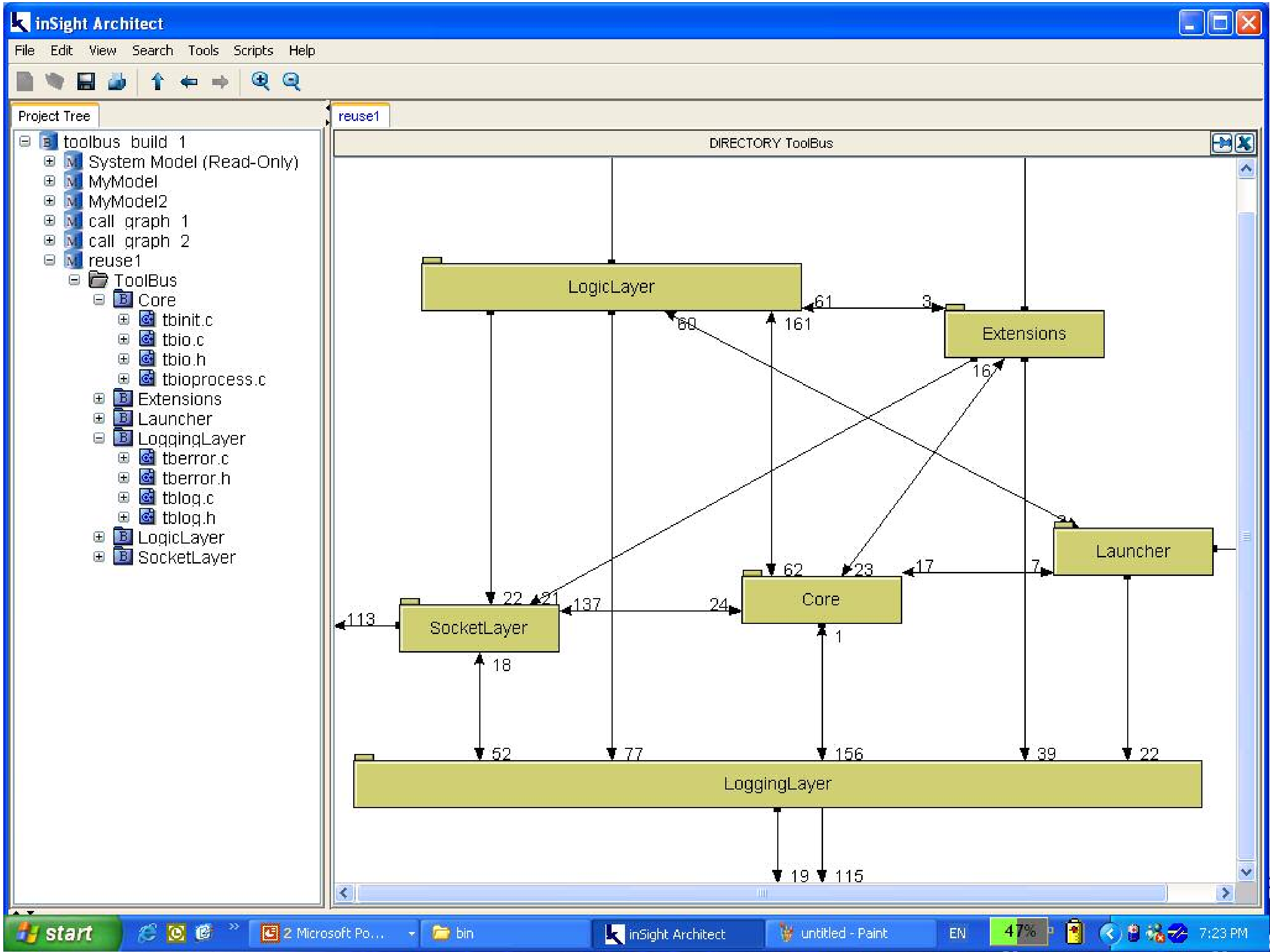


inSight Architect
File Edit View Search Tools Scripts Help
Project Tree
toolbus build 1
System Model (Read-Only)
MyModel
MyModel2
call graph 1
call graph 2
reuse1
ToolBus
accept.h
binbuffer.c
binbuffer.h
config.h
select.h
socketio.c
socketio.h
stringtoid.c
stringtoid.h
tbcommunication.c
tbcommunication.h
tbconnection.c
tbconnection.h
tbcontrol.c
tbcontrol.h
tberror.c
tberror.h
tbevents.c
tbextensions.h
tbinit.c
tbio.c
tbio.h
tbioprocess.c
tblauncher.c
tblauncher.h
tblog.c
tblog.h
tbmenu.c
tbmessages.h
tbtoollist.c
reuse1
DIRECTORY ToolBus
accept.h
binbuffer.c
binbuffer.h
config.h
socketio.c
socketio.h
stringtoid.c
stringtoid.h
tbcommunication...
tbconnection.c
tbconnection.h
tbcontrol.c
tbevents.c
tbextensions.h
tbio.c
tbio.h
tbioprocess.c
tblauncher.c
tbmenu.c
tbmessages.h
tbtoollist.h
tbtypes.h
version.h
tblog.c
tblog.h
tberror.h
tberror.c
start
2 Microsoft Po...
bin
inSight Architect
untitled - Paint
EN
50%
7:16 PM

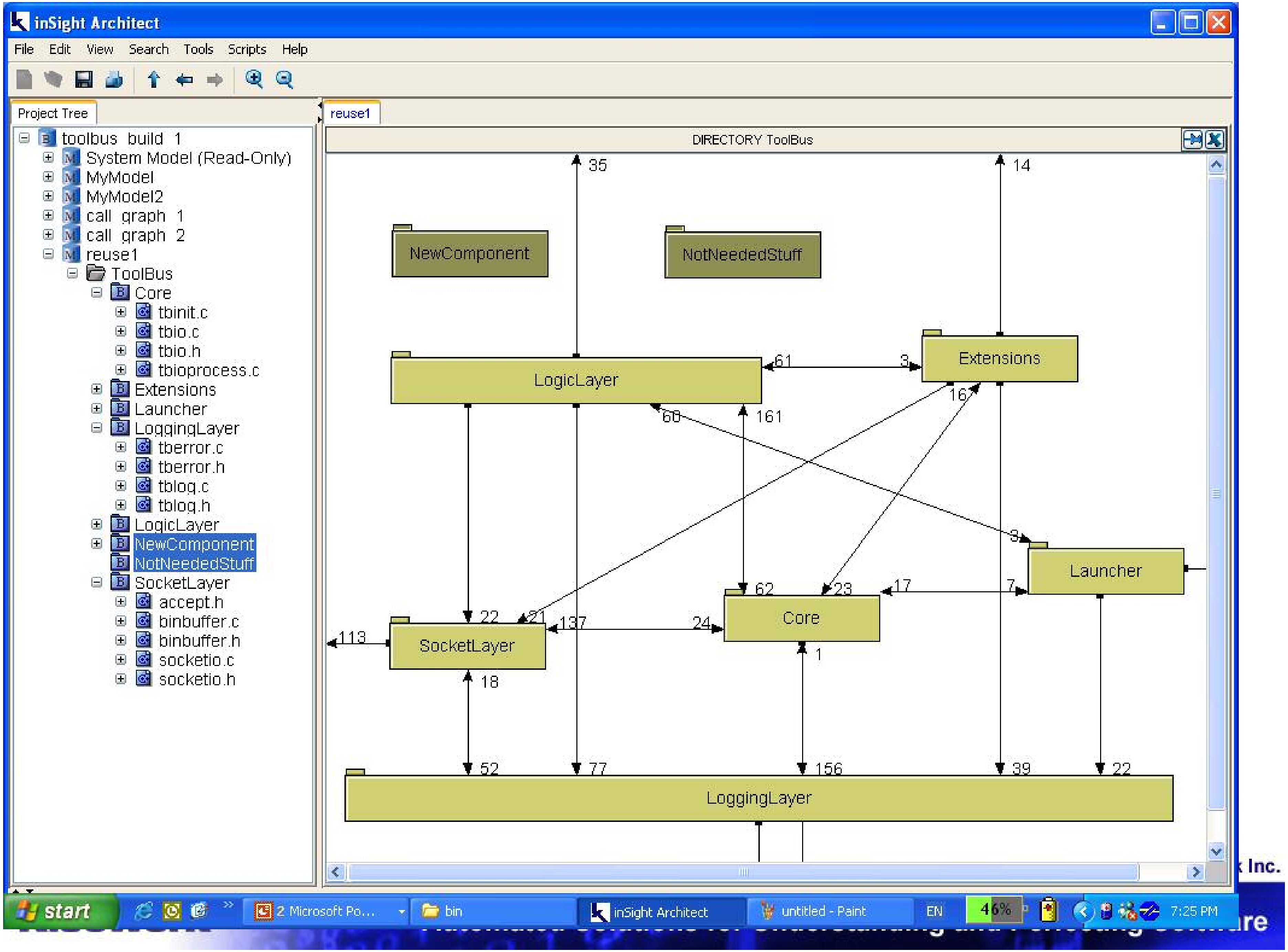


inSight Architect
File Edit View Search Tools Scripts Help
Project Tree
toolbus build 1
System Model (Read-Only)
MyModel
MyModel2
call graph 1
call graph 2
reuse1
ToolBus
LoggingLayer
accept.h
binbuffer.c
binbuffer.h
config.h
select.h
socketio.c
socketio.h
stringtoid.c
stringtoid.h
tbcommunication.c
tbcommunication.h
tbconnection.c
tbconnection.h
tbcontrol.c
tbcontrol.h
tbevents.c
tbextensions.h
tbinit.c
tbio.c
tbio.h
tbioprocess.c
tblauncher.c
tblauncher.h
tbmenu.c
tbmessages.h
tbtoollist.c
tbtoollist.h
tbtypes.h
version.h
reuse1
DIRECTORY ToolBus
accept.h
binbuffer.c
binbuffer.h
config.h
socketio.c
socketio.h
stringtoid.c
stringtoid.h
tbcommunication...
tbconnection.c
tbconnection.h
tbcontrol.c
tbevents.c
tbextensions.h
tbio.c
tbio.h
tbioprocess.c
tblauncher.c
tbmenu.c
tbmessages.h
tbtoollist.h
tbtypes.h
version.h
LoggingLayer
start
2 Microsoft Po...
bin
inSight Architect
untitled - Paint
EN
49%
7:18 PM

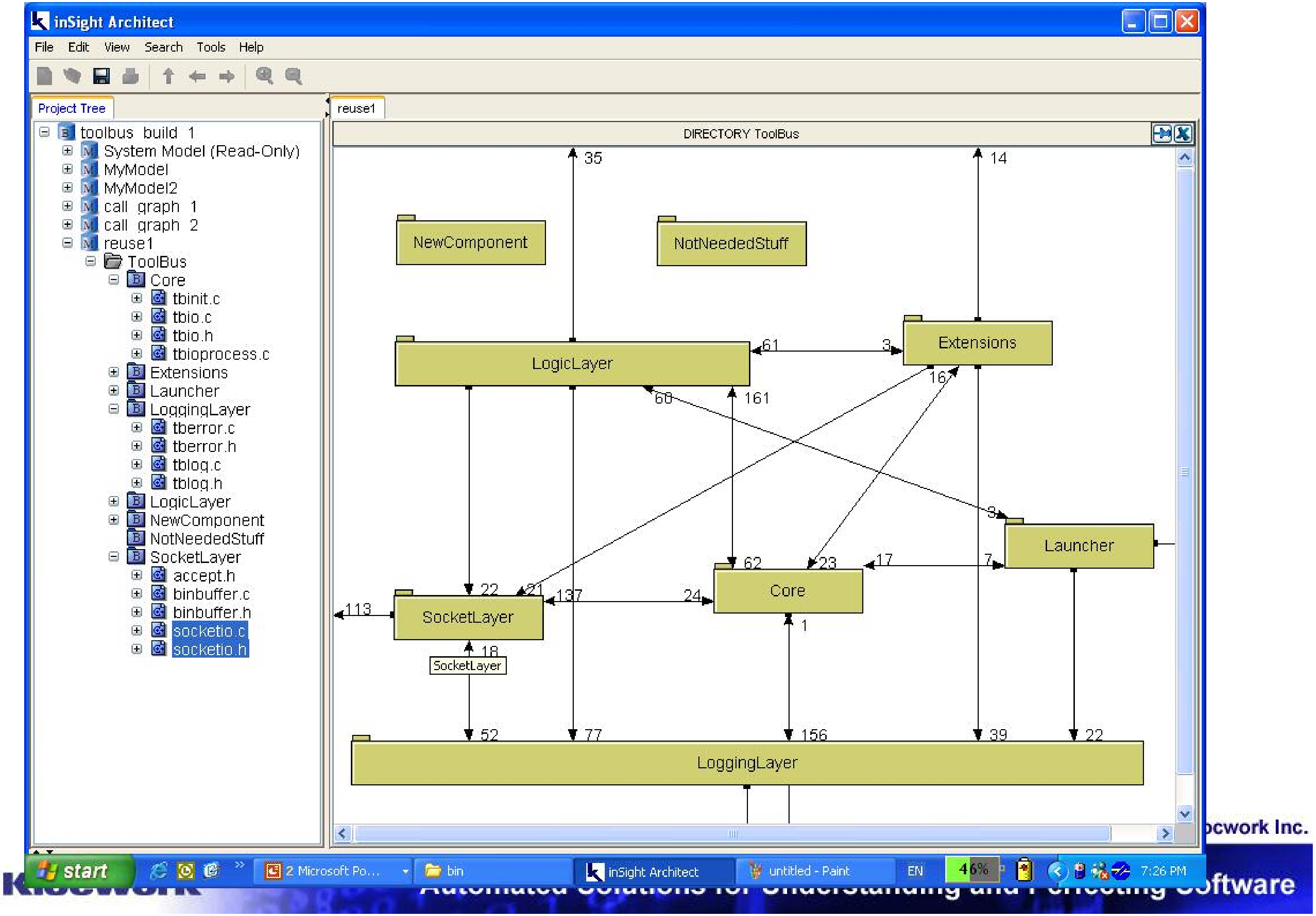

inSight Architect
File Edit View Search Tools Scripts Help
Project Tree
toolbus build 1
System Model (Read-Only)
MyModel
MyModel2
call graph 1
call graph 2
reuse1
ToolBus
LoggingLayer
tberror.c
tberror.h
tblog.c
tblog.h
accept.h
binbuffer.c
binbuffer.h
config.h
select.h
socketio.c
socketio.h
stringtoid.c
stringtoid.h
tbcommunication.c
tbcommunication.h
tbconnection.c
tbconnection.h
tbcontrol.c
tbcontrol.h
tbevents.c
tbextensions.h
tbinit.c
tbio.c
tbio.h
tbioprocess.c
tblauncher.c
tblauncher.h
tbmenu.c
tbmessages.h
reuse1
ARCHITECTURE-BLOCK LoggingLayer
80
tberror.h
7
tblog.c
7
3
1
tberror.c
29
13
5
tblog.h
18
234
1
rk Inc.
are
start
2 Microsoft Po...
bin
inSight Architect
untitled - Paint
EN
49%
7:19 PM

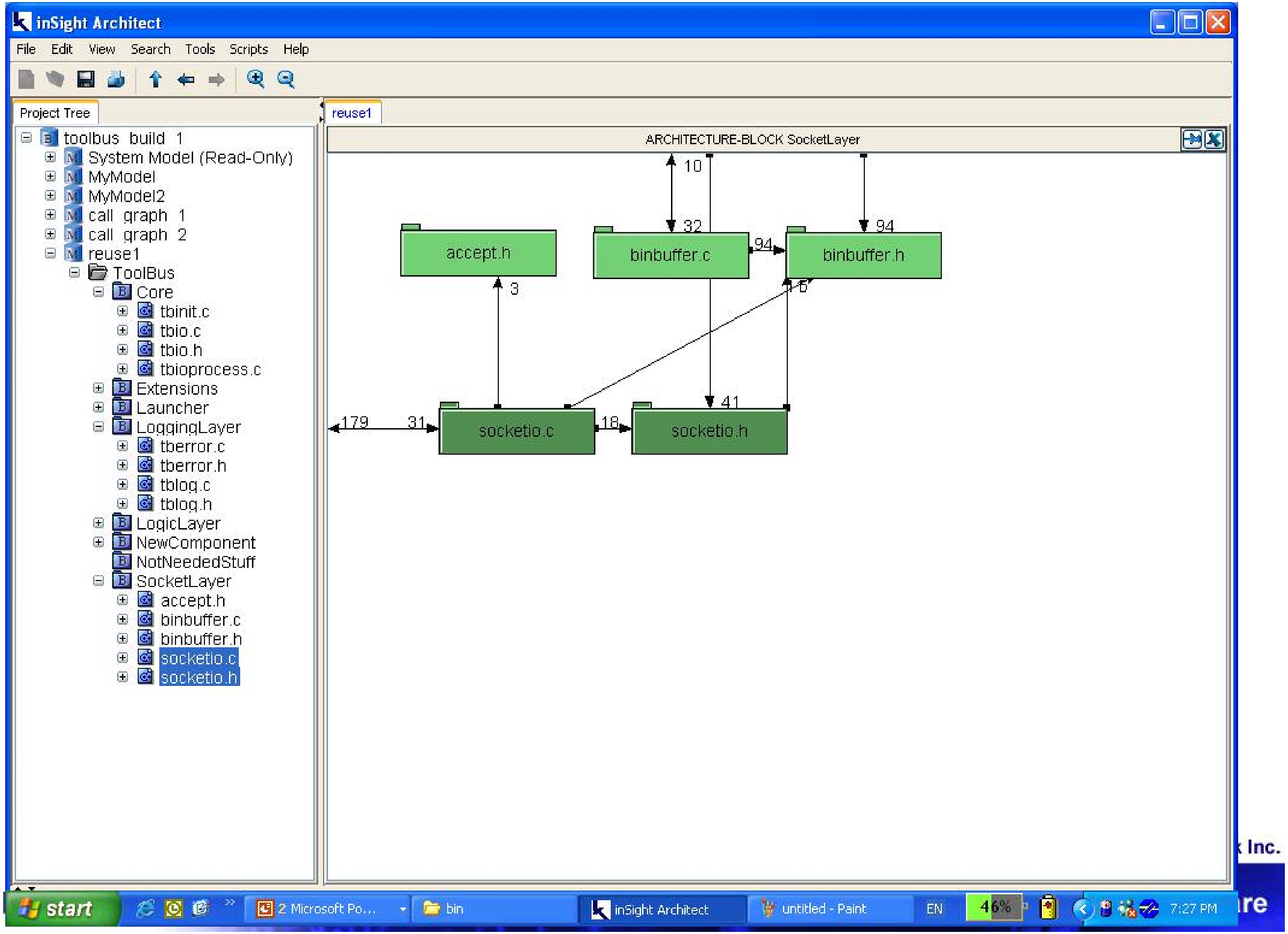


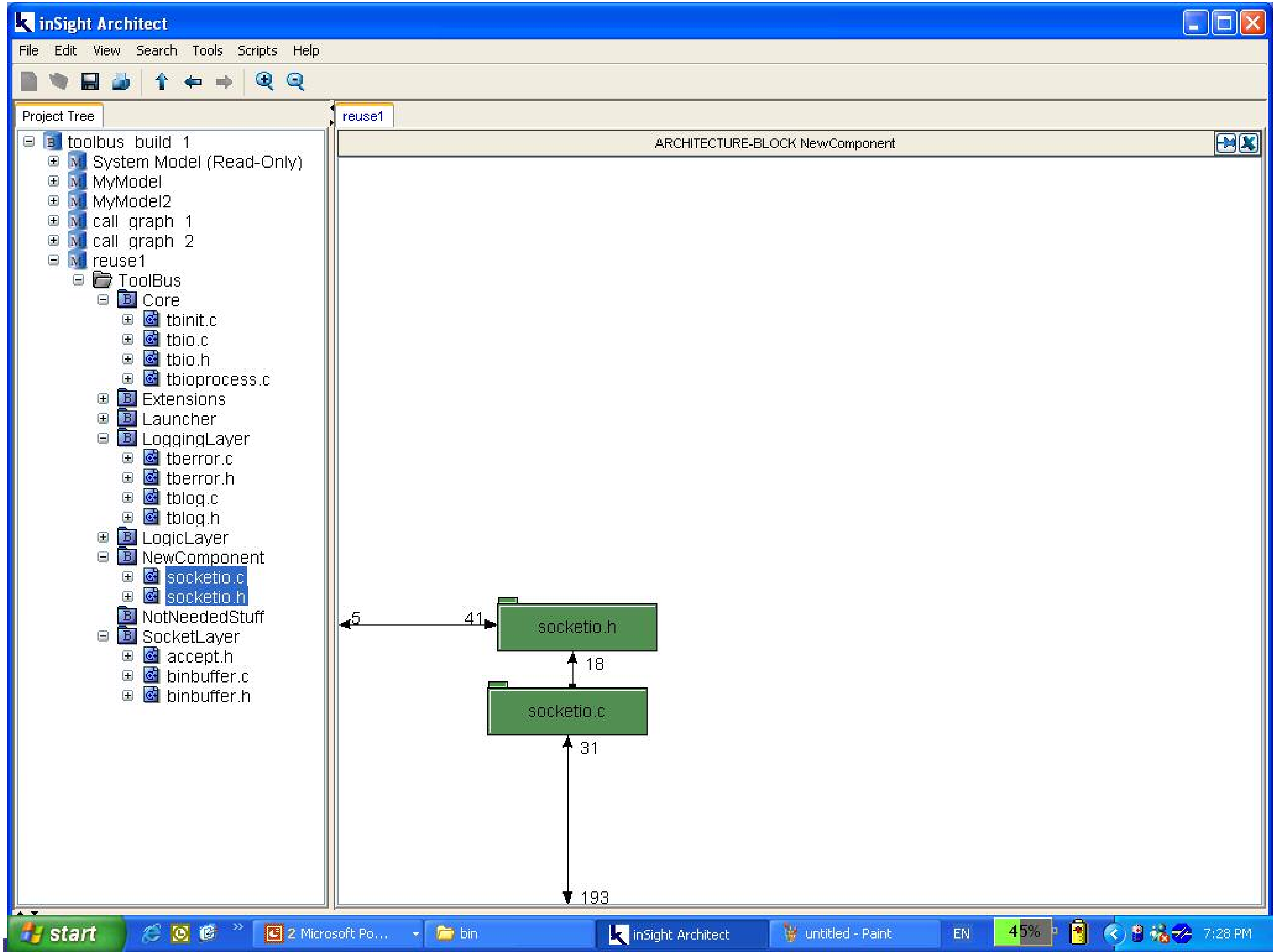
ocwork Inc.
oftware

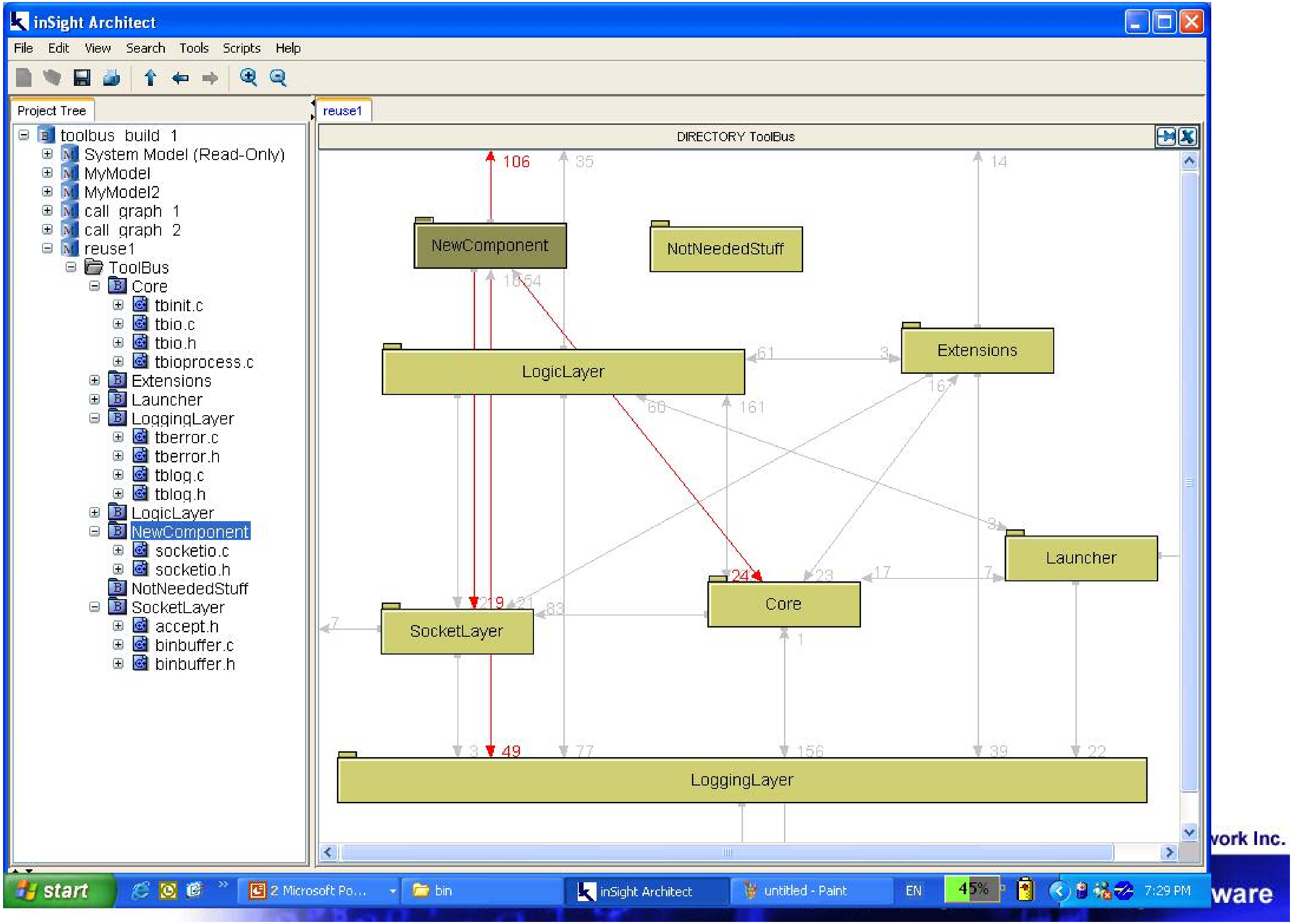


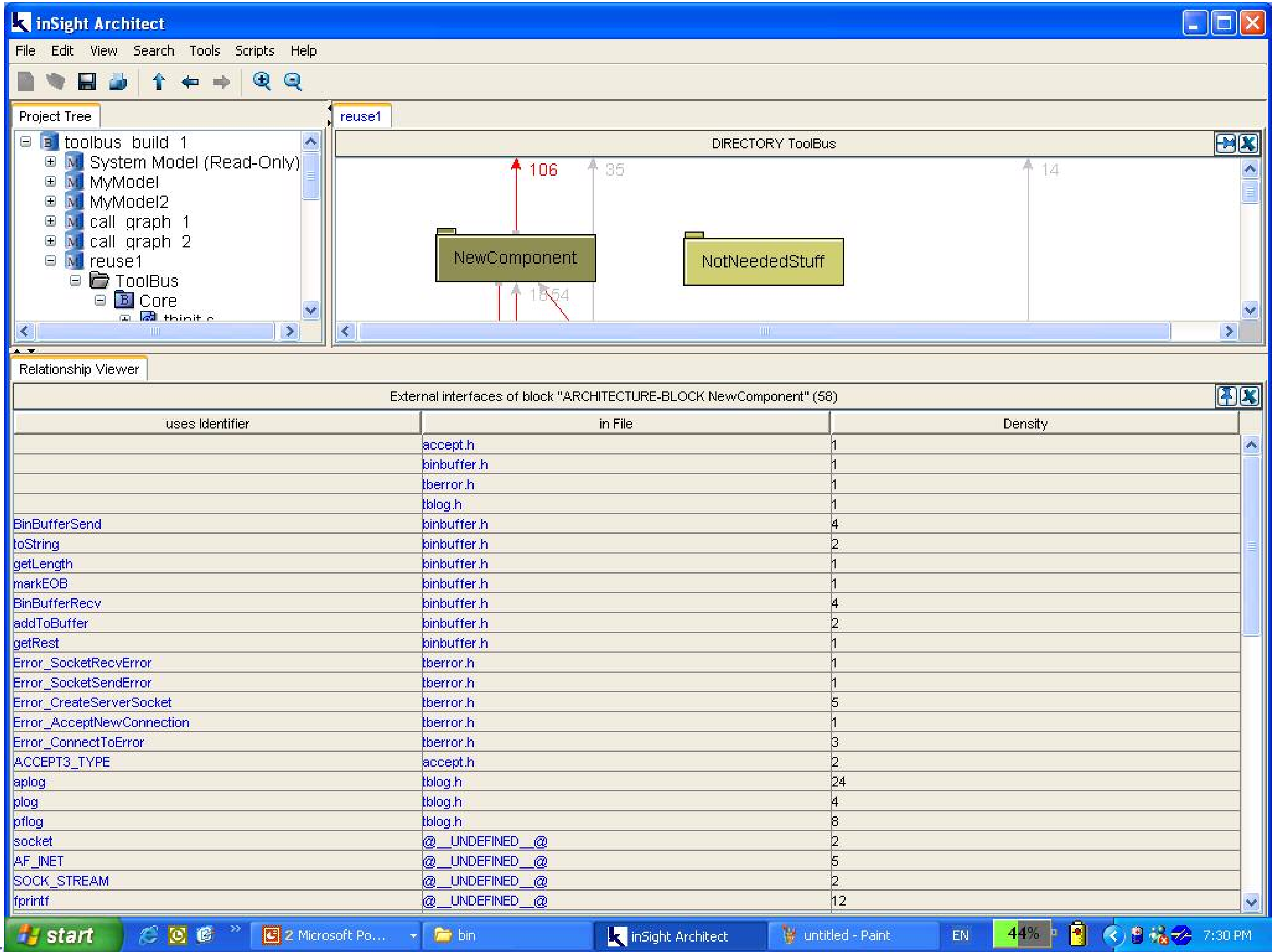
inSight Architect
File Edit View Search Tools Scripts Help
Project Tree
reuse1
DIRECTORY ToolBus
toolbus build 1
System Model (Read-Only)
MyModel
MyModel2
call graph 1
call graph 2
reuse1
ToolBus
Core
tbinit.c
tbio.c
tbio.h
tbioprocess.c
Extensions
Launcher
LoggingLayer
tberror.c
tberror.h
tblog.c
tblog.h
LogicLayer
NewComponent
NotNeededStuff
SocketLayer
accept.h
binbuffer.c
binbuffer.h
socketio.c
socketio.h
NewComponent
NotNeededStuff
LogicLayer
Extensions
Launcher
Core
SocketLayer
LoggingLayer
35
14
61
3
16
60
161
3
62
23
17
7
22
21
137
24
113
1
18
52
77
156
39
22
start
2 Microsoft Po...
bin
inSight Architect
untitled - Paint
EN
46%
7:25 PM


inSight Architect
File Edit View Search Tools Help
Project Tree
toolbus build 1
System Model (Read-Only)
MyModel
MyModel2
call graph 1
call graph 2
reuse1
ToolBus
Core
tbinit.c
tbio.c
tbio.h
tbioprocess.c
Extensions
Launcher
LoggingLayer
tberror.c
tberror.h
tblog.c
tblog.h
LogicLayer
NewComponent
NotNeededStuff
SocketLayer
accept.h
binbuffer.c
binbuffer.h
socketio.c
socketio.h
reuse1
DIRECTORY ToolBus
NewComponent
NotNeededStuff
Extensions
LogicLayer
Launcher
Core
SocketLayer
LoggingLayer
start
2 Microsoft Po...
bin
inSight Architect
untitled - Paint
EN
7:26 PM
ocwork Inc.
Automated Solutions for Understanding and Perfecting Software

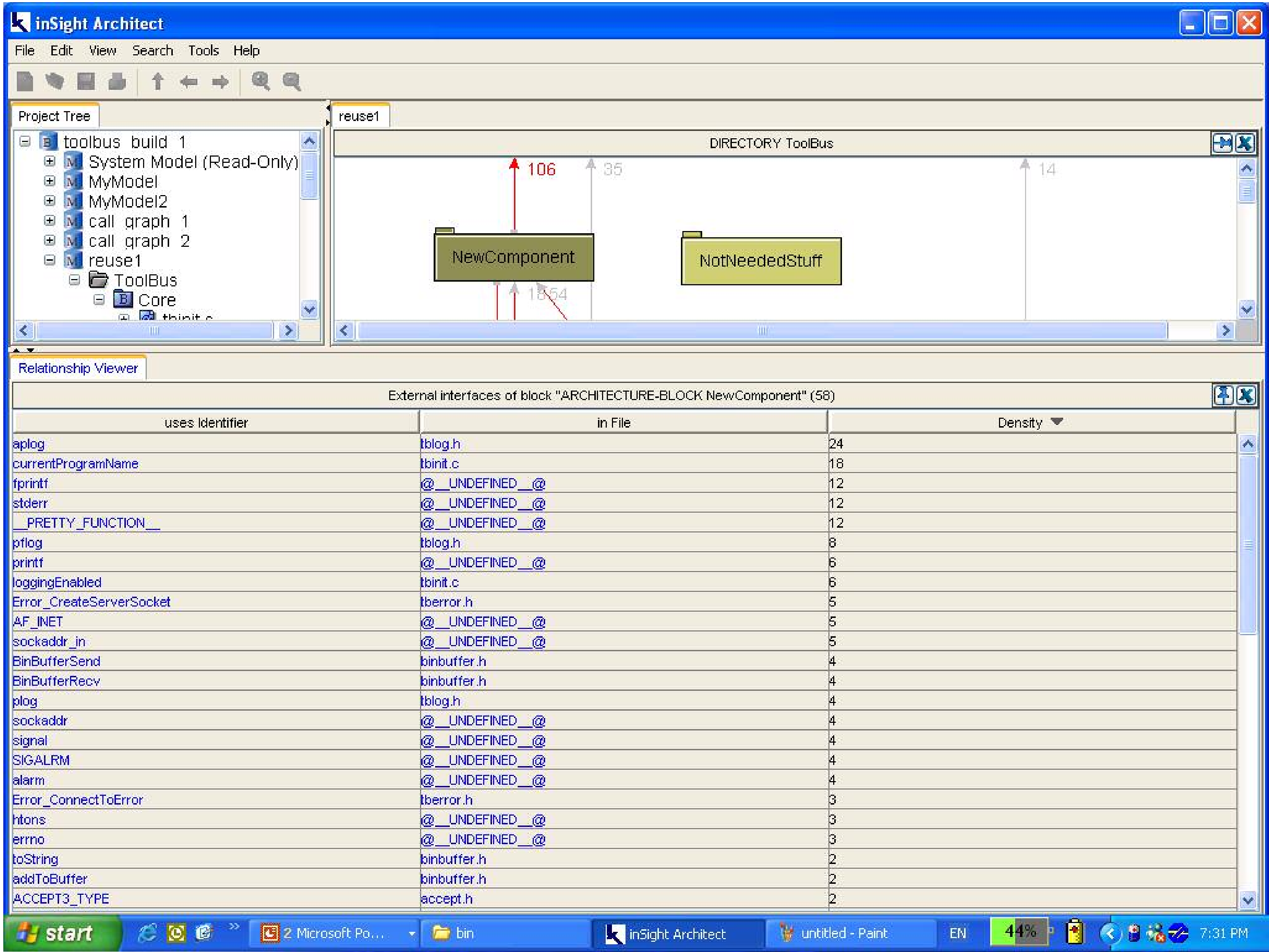

inSight Architect
File Edit View Search Tools Scripts Help
Project Tree
reuse1
toolbus build 1
System Model (Read-Only)
MyModel
MyModel2
call graph 1
call graph 2
reuse1
ToolBus
Core
tbinit.c
tbio.c
tbio.h
tbioprocess.c
Extensions
Launcher
LoggingLayer
tberror.c
tberror.h
tblog.c
tblog.h
LogicLayer
NewComponent
NotNeededStuff
SocketLayer
accept.h
binbuffer.c
binbuffer.h
socketio.c
socketio.h
ARCHITECTURE-BLOCK SocketLayer
10
32
94
94
accept.h
binbuffer.c
binbuffer.h
3
179
31
18
41
socketio.c
socketio.h
start
2 Microsoft Po...
bin
inSight Architect
untitled - Paint
EN
46%
7:27 PM

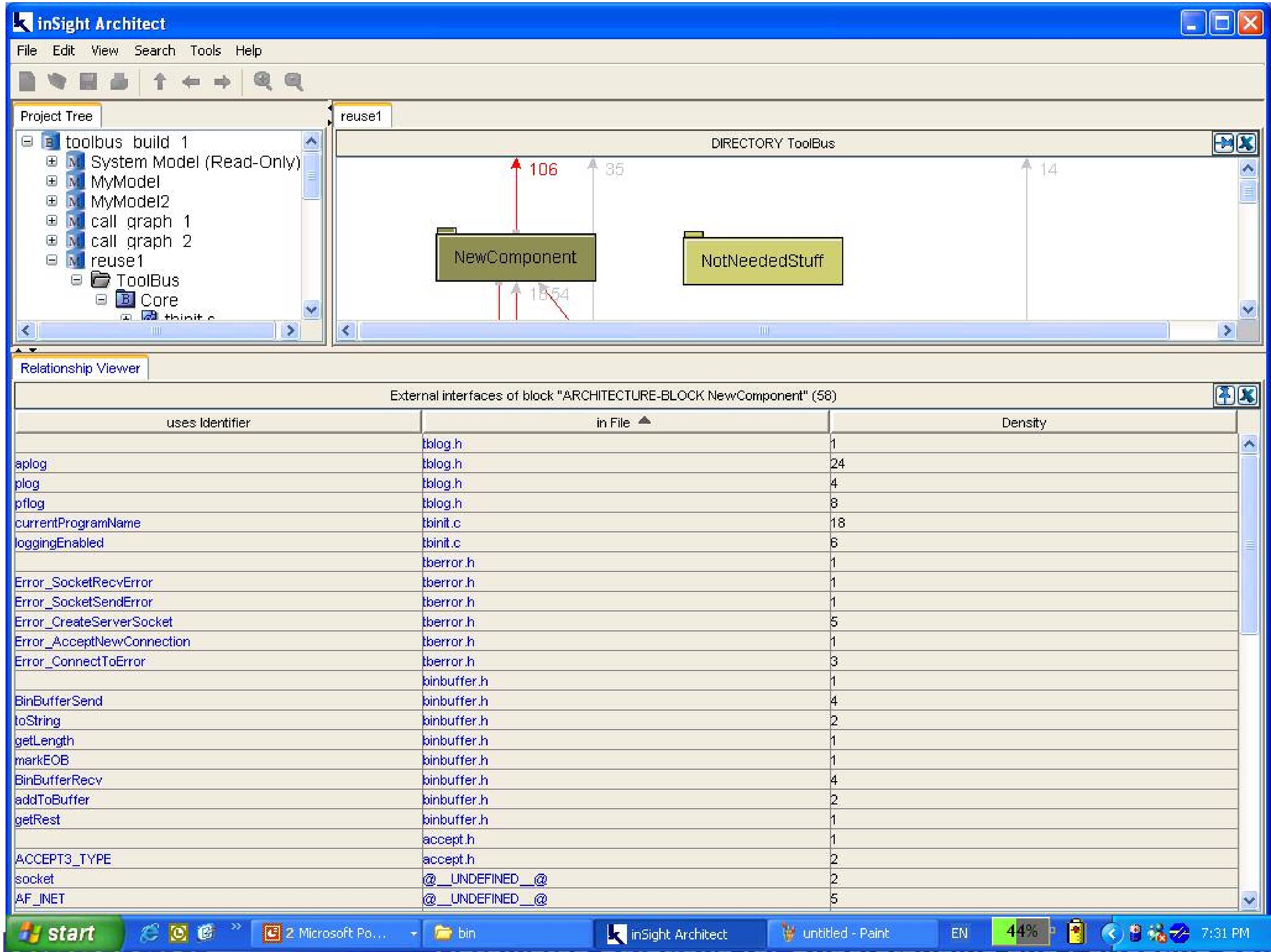

inSight Architect
File Edit View Search Tools Scripts Help
Project Tree
toolbus build 1
System Model (Read-Only)
MyModel
MyModel2
call graph 1
call graph 2
reuse1
ToolBus
Core
tbinit.c
tbio.c
tbio.h
tbioprocess.c
Extensions
Launcher
LoggingLayer
tberror.c
tberror.h
tblog.c
tblog.h
LogicLayer
NewComponent
socketio.c
socketio.h
NotNeededStuff
SocketLayer
accept.h
binbuffer.c
binbuffer.h
reuse1
ARCHITECTURE-BLOCK NewComponent
5
41
socketio.h
18
socketio.c
31
193
start
2 Microsoft Po...
bin
inSight Architect
untitled - Paint
EN
45%
7:28 PM


inSight Architect
File Edit View Search Tools Scripts Help
Project Tree
toolbus build 1
System Model (Read-Only)
MyModel
MyModel2
call graph 1
call graph 2
reuse1
ToolBus
Core
tbinit.c
tbio.c
tbio.h
tbioprocess.c
Extensions
Launcher
LoggingLayer
tberror.c
tberror.h
tblog.c
tblog.h
LogicLayer
NewComponent
socketio.c
socketio.h
NotNeededStuff
SocketLayer
accept.h
binbuffer.c
binbuffer.h
reuse1
DIRECTORY ToolBus
NewComponent
NotNeededStuff
LogicLayer
Extensions
Launcher
Core
SocketLayer
LoggingLayer
106
35
14
61
3
16
60
161
24
23
17
7
219
83
1
3
49
77
156
39
22
start
2 Microsoft Po...
bin
inSight Architect
untitled - Paint
EN
45%
7:29 PM
vork Inc.
ware

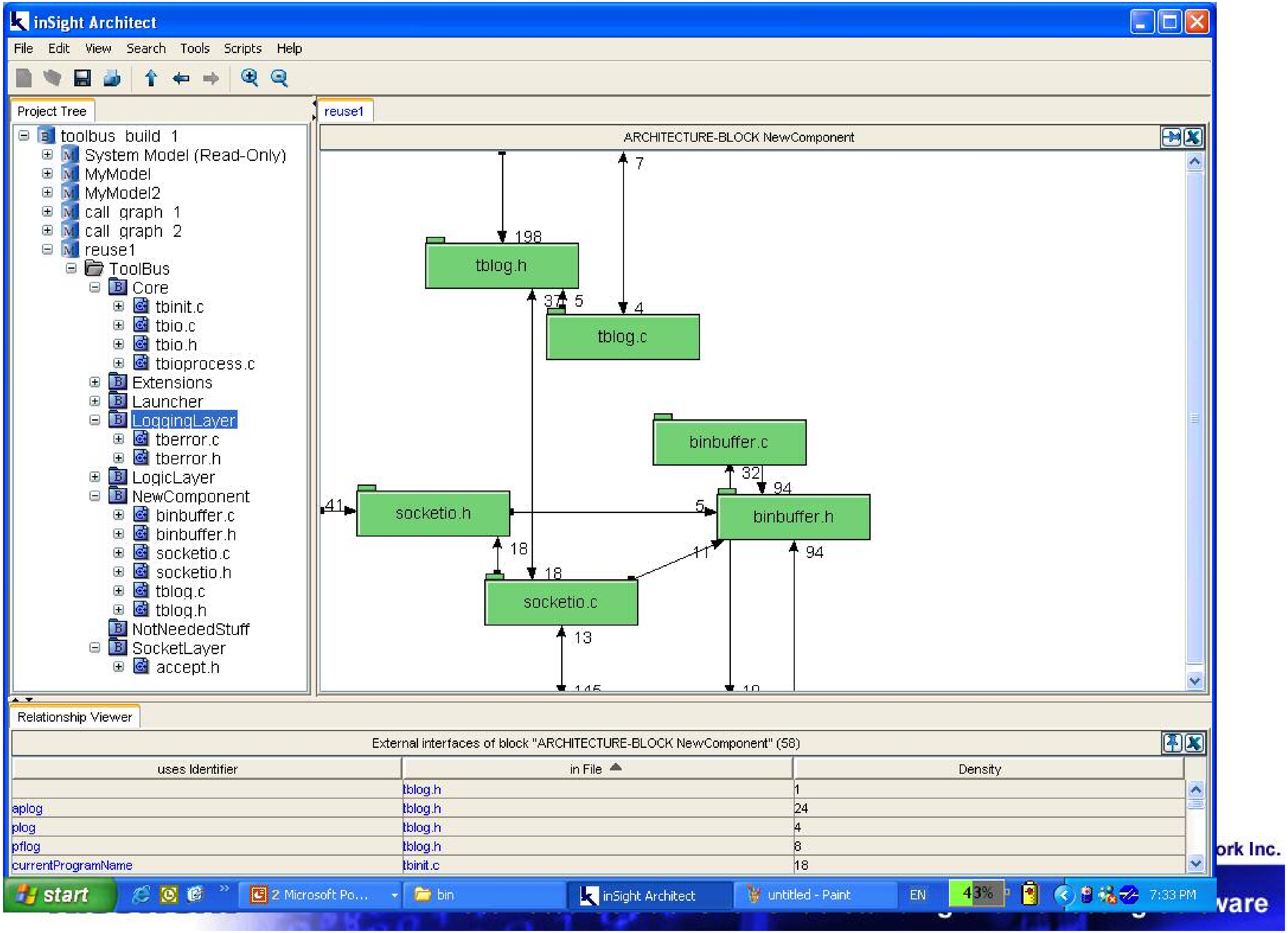


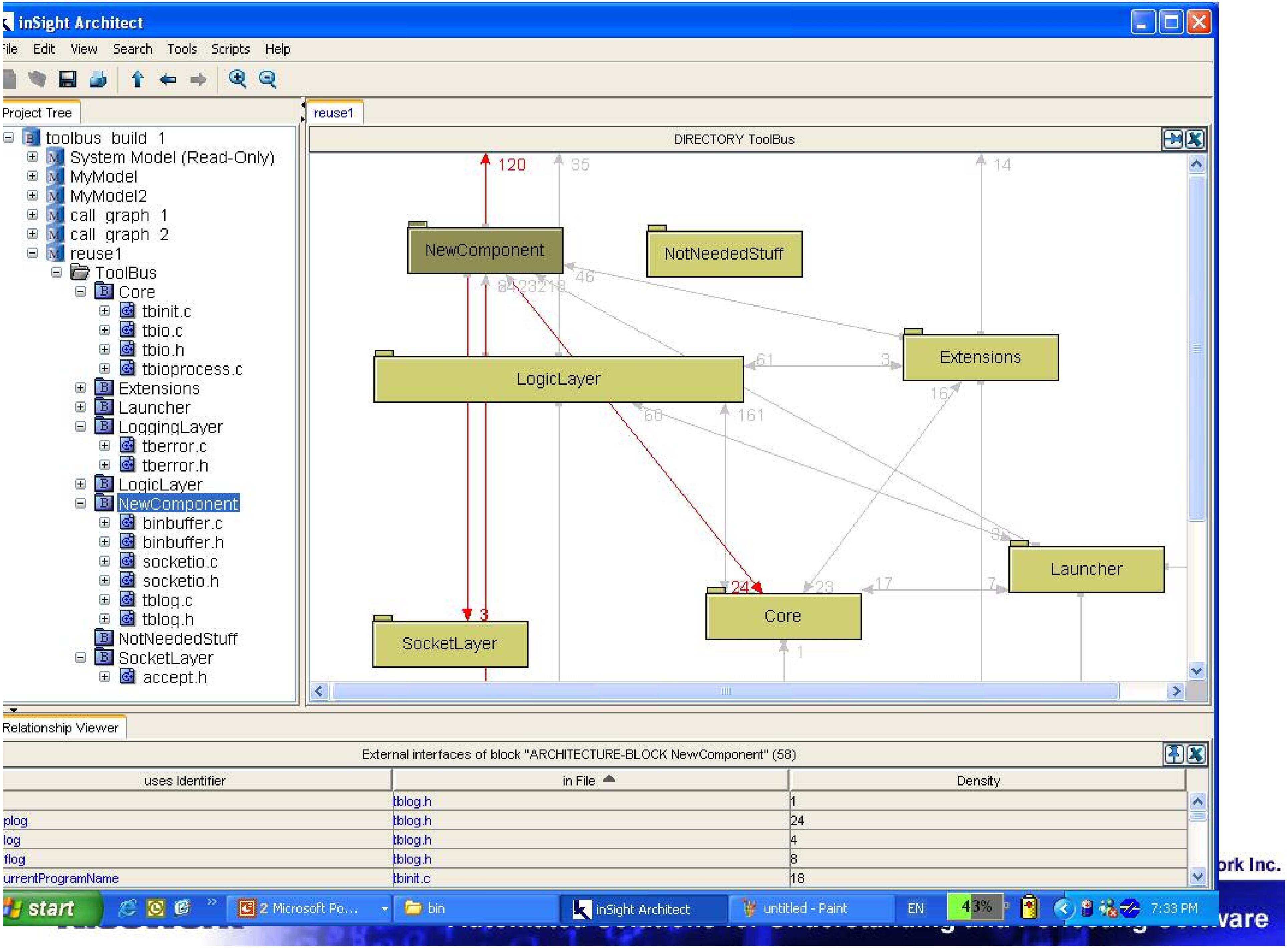
uses Identifier	in File	Density
	accept.h	1
	binbuffer.h	1
	tberror.h	1
	tblog.h	1
BinBufferSend	binbuffer.h	4
toString	binbuffer.h	2
getLength	binbuffer.h	1
markEOB	binbuffer.h	1
BinBufferRecv	binbuffer.h	4
addToBuffer	binbuffer.h	2
getRest	binbuffer.h	1
Error_SocketRecvError	tberror.h	1
Error_SocketSendError	tberror.h	1
Error_CreateServerSocket	tberror.h	5
Error_AcceptNewConnection	tberror.h	1
Error_ConnectToError	tberror.h	3
ACCEPT3_TYPE	accept.h	2
aplog	tblog.h	24
plog	tblog.h	4
pflog	tblog.h	8
socket	@__UNDEFINED__@	2
AF_INET	@__UNDEFINED__@	5
SOCK_STREAM	@__UNDEFINED__@	2
fprintf	@__UNDEFINED__@	12


inSight Architect
File Edit View Search Tools Help
Project Tree
toolbus build 1
System Model (Read-Only)
MyModel
MyModel2
call graph 1
call graph 2
reuse1
ToolBus
Core
reuse1
DIRECTORY ToolBus
106
35
14
NewComponent
NotNeededStuff
Relationship Viewer
External interfaces of block "ARCHITECTURE-BLOCK NewComponent" (58)
uses Identifier | in File | Density
aplog | tblog.h | 24
currentProgramName | tbinit.c | 18
fprintf | @__UNDEFINED__@ | 12
stderr | @__UNDEFINED__@ | 12
__PRETTY_FUNCTION__ | @__UNDEFINED__@ | 12
pflog | tblog.h | 8
printf | @__UNDEFINED__@ | 6
loggingEnabled | tbinit.c | 6
Error_CreateServerSocket | tberror.h | 5
AF_INET | @__UNDEFINED__@ | 5
sockaddr_in | @__UNDEFINED__@ | 5
BinBufferSend | binbuffer.h | 4
BinBufferRecv | binbuffer.h | 4
plog | tblog.h | 4
sockaddr | @__UNDEFINED__@ | 4
signal | @__UNDEFINED__@ | 4
SIGALRM | @__UNDEFINED__@ | 4
alarm | @__UNDEFINED__@ | 4
Error_ConnectToError | tberror.h | 3
htons | @__UNDEFINED__@ | 3
errno | @__UNDEFINED__@ | 3
toString | binbuffer.h | 2
addToBuffer | binbuffer.h | 2
ACCEPT3_TYPE | accept.h | 2
start
2 Microsoft Po...
bin
inSight Architect
untitled - Paint
EN
44%
7:31 PM

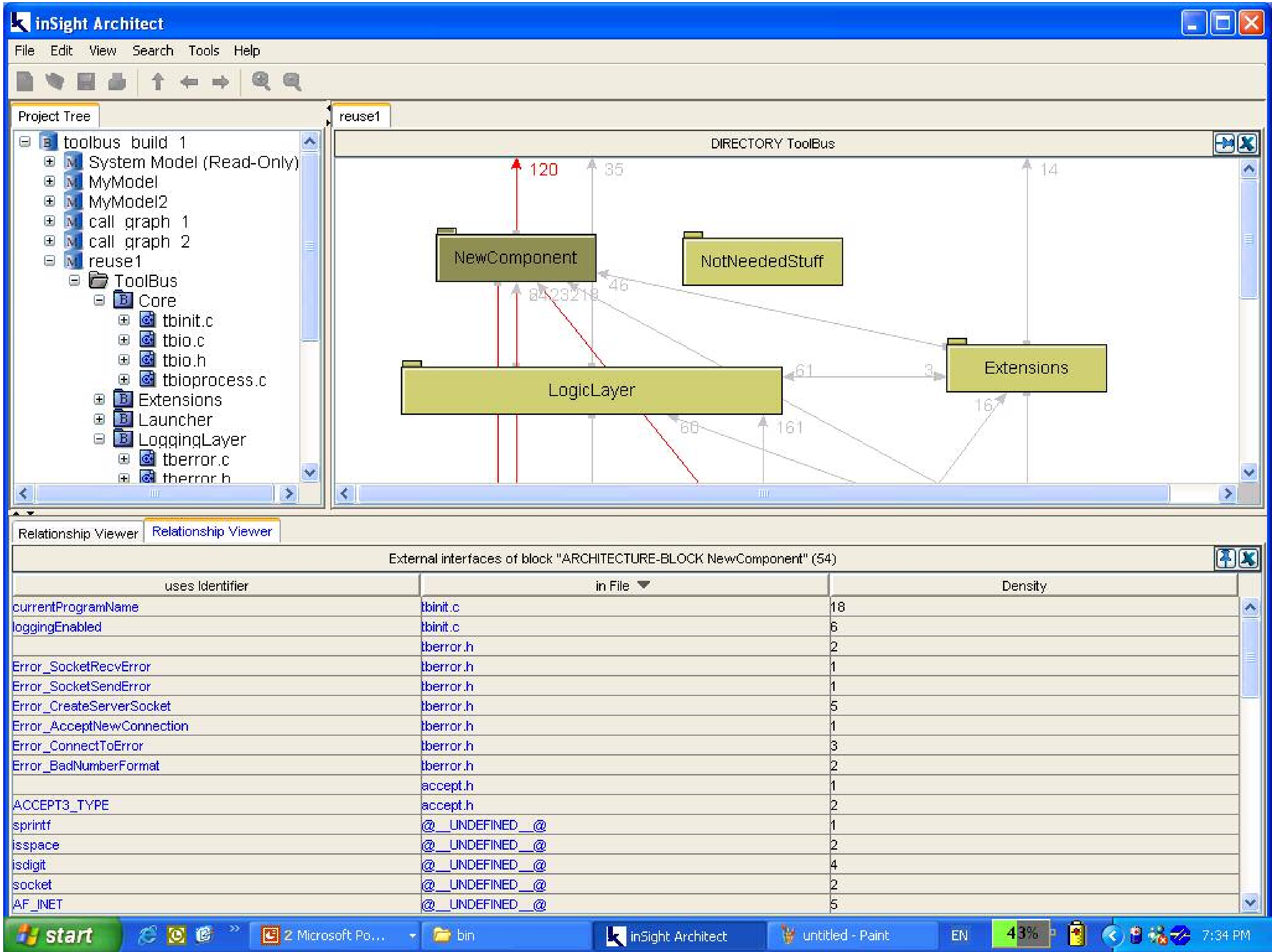


uses Identifier	in File	Density
	tblog.h	1
aplog	tblog.h	24
plog	tblog.h	4
pflog	tblog.h	8
currentProgramName	tbinit.c	18
loggingEnabled	tbinit.c	6
	tberror.h	1
Error_SocketRecvError	tberror.h	1
Error_SocketSendError	tberror.h	1
Error_CreateServerSocket	tberror.h	5
Error_AcceptNewConnection	tberror.h	1
Error_ConnectToError	tberror.h	3
	binbuffer.h	1
BinBufferSend	binbuffer.h	4
toString	binbuffer.h	2
getLength	binbuffer.h	1
markEOB	binbuffer.h	1
BinBufferRecv	binbuffer.h	4
addToBuffer	binbuffer.h	2
getRest	binbuffer.h	1
	accept.h	1
ACCEPT3_TYPE	accept.h	2
socket	@__UNDEFINED__@	2
AF_INET	@__UNDEFINED__@	5


inSight Architect
File Edit View Search Tools Scripts Help
Project Tree
toolbus build 1
System Model (Read-Only)
MyModel
MyModel2
call graph 1
call graph 2
reuse1
ToolBus
Core
tbinit.c
tbio.c
tbio.h
tbioprocess.c
Extensions
Launcher
LoggingLayer
tberror.c
tberror.h
LogicLayer
NewComponent
binbuffer.c
binbuffer.h
socketio.c
socketio.h
tblog.c
tblog.h
NotNeededStuff
SocketLayer
accept.h
reuse1
ARCHITECTURE-BLOCK NewComponent
tblog.h
tblog.c
binbuffer.c
binbuffer.h
socketio.h
socketio.c
198
7
37
5
4
32
94
41
5
18
11
18
94
13
Relationship Viewer
External interfaces of block "ARCHITECTURE-BLOCK NewComponent" (58)
uses Identifier
in File
Density
tblog.h
1
aplog
tblog.h
24
plog
tblog.h
4
pflog
tblog.h
8
currentProgramName
tbinit.c
18
start
2 Microsoft Po...
bin
inSight Architect
untitled - Paint
EN
43%
7:33 PM
ork Inc.
vare



inSight Architect
File Edit View Search Tools Scripts Help
Project Tree
toolbus build 1
System Model (Read-Only)
MyModel
MyModel2
call graph 1
call graph 2
reuse1
ToolBus
Core
tbinit.c
tbio.c
tbio.h
tbioprocess.c
Extensions
Launcher
LoggingLayer
tberror.c
tberror.h
LogicLayer
NewComponent
binbuffer.c
binbuffer.h
socketio.c
socketio.h
tblog.c
tblog.h
NotNeededStuff
SocketLayer
accept.h
reuse1
DIRECTORY ToolBus
NewComponent
NotNeededStuff
LogicLayer
Extensions
Launcher
Core
SocketLayer
120
35
14
46
61
3
16
68
161
24
23
17
7
1
Relationship Viewer
External interfaces of block "ARCHITECTURE-BLOCK NewComponent" (58)
uses Identifier
in File
Density
tblog.h 1
plog tblog.h 24
log tblog.h 4
flog tblog.h 8
urrentProgramName tbinit.c 18
start
2 Microsoft Po...
bin
inSight Architect
untitled - Paint
EN
7:33 PM
ork Inc.
vare


inSight Architect
File Edit View Search Tools Help
Project Tree
toolbus build 1
System Model (Read-Only)
MyModel
MyModel2
call graph 1
call graph 2
reuse1
ToolBus
Core
tbinit.c
tbio.c
tbio.h
tbioprocess.c
Extensions
Launcher
LoggingLayer
tberror.c
tberror.h
reuse1
DIRECTORY ToolBus
120
35
14
NewComponent
NotNeededStuff
46
Extensions
LogicLayer
61
3
16
68
161
Relationship Viewer
Relationship Viewer
External interfaces of block "ARCHITECTURE-BLOCK NewComponent" (54)
uses Identifier | in File | Density
currentProgramName | tbinit.c | 18
loggingEnabled | tbinit.c | 6
| tberror.h | 2
Error_SocketRecvError | tberror.h | 1
Error_SocketSendError | tberror.h | 1
Error_CreateServerSocket | tberror.h | 5
Error_AcceptNewConnection | tberror.h | 1
Error_ConnectToError | tberror.h | 3
Error_BadNumberFormat | tberror.h | 2
| accept.h | 1
ACCEPT3_TYPE | accept.h | 2
sprintf | @__UNDEFINED__@ | 1
isspace | @__UNDEFINED__@ | 2
isdigit | @__UNDEFINED__@ | 4
socket | @__UNDEFINED__@ | 2
AF_INET | @__UNDEFINED__@ | 5
start
2 Microsoft Po...
bin
inSight Architect
untitled - Paint
EN
43%
7:34 PM

Conclusions/Key points

- **Reuse activities add another dimension to multi-release software production**
  - Reuse for maintenance (cloning)
  - Reuse for modernization
  - Reuse for redevelopment
  - Reuse for product line (across products)
- **Reuse is an architecture-centric activity**
- **Suitable Architecture Models are required to manage reuse**
- **Harvesting is an architecture refactoring**
  - First, you need to excavate your architecture, as a precise model
  - Harvesting requires the following steps:
    - Identify/localize/collect
    - Isolate component
    - Introduce a boundary
    - Implement refactoring