Dependency Analysis for Managing Structural Complexity - Android case

2010.11.4
Chanjin Park

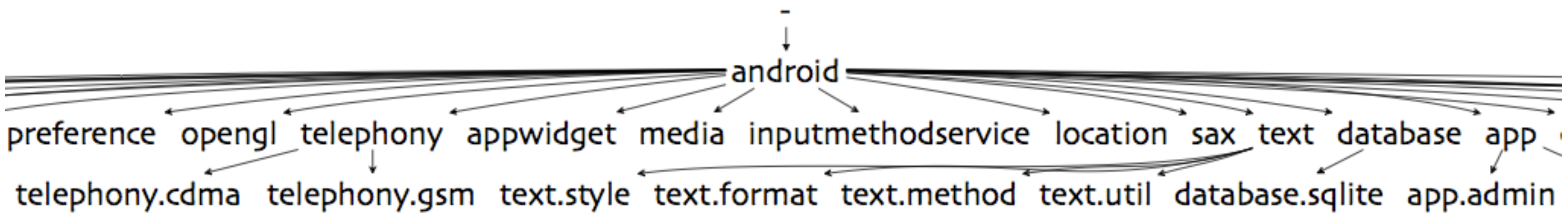
Topics

- *Why manage structure?*
- *Package Layering*
  - Case Study - Android

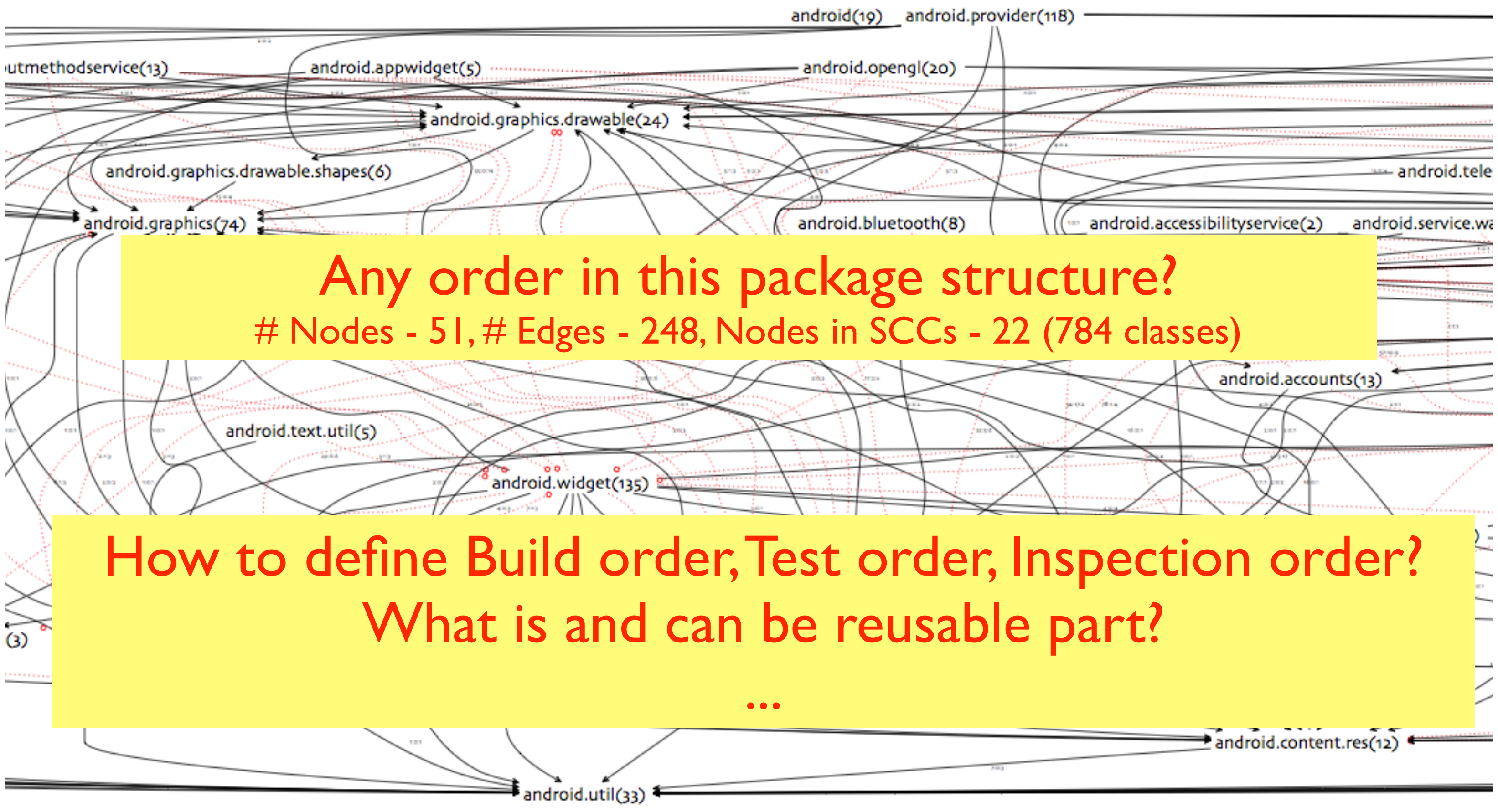
Why manage structure?

Why manage package structure?

- *Divide & Conquer*
  - Development unit, Work assignment (Divide works and integrate later)
  - Reuse unit, Release unit, Testing unit
  - Mostly divide software with functional constructs such as funcitons, files and directories
- *Package hierarchy*
  - Directory structure
  - Version control operation - Checkin, Checkout, ...



- *Package dependency structure*
  - Usage relation among classes in packages: Import
  - Packages needed to implement



Any order in this package structure?

# Nodes - 51, # Edges - 248, Nodes in SCCs - 22 (784 classes)

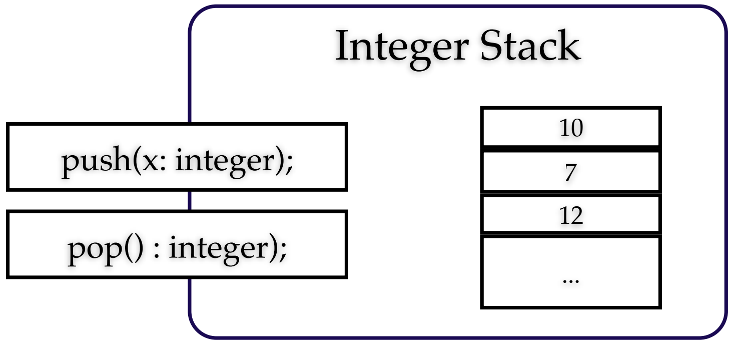
How to define Build order, Test order, Inspection order?

What is and can be reusable part?

...

Some principles for architecture

- *Information hiding*
  - Hide complex and easily changeable things for modifying independently without affecting other things
    - Stable Interface and Extensive Changeable Implementation details
  - Encapsulation: Private and public
  - Abstraction: Hiding details



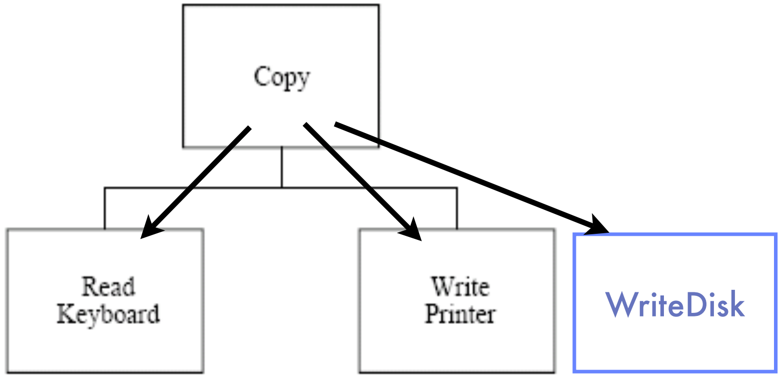
Linked List, Array, ...

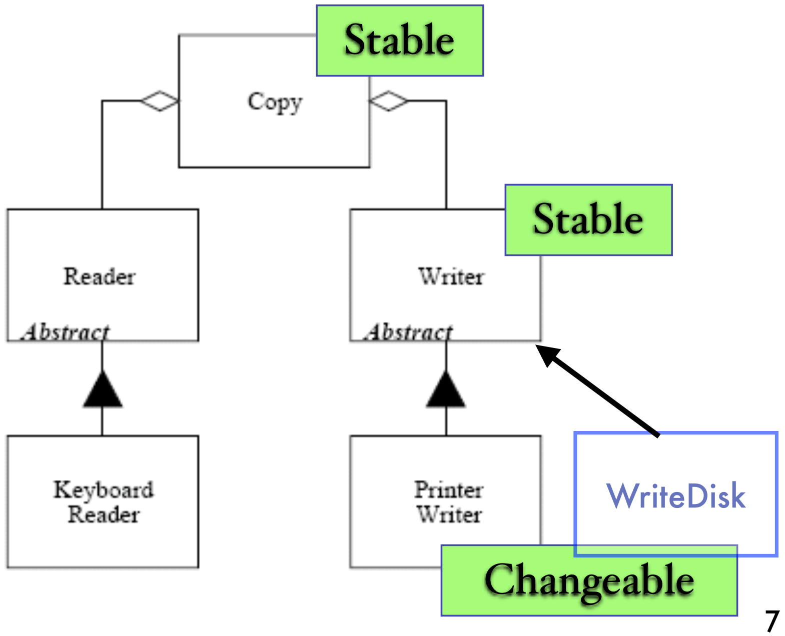
- ***Separation of concerns***
  - Traditionally, each separated units are mapped to Class, File, Function, ...
  - Aspect Oriented Programming

- ***Dependency Inversion***
  - Dependency should be from Changeable to Stable and from Concrete to Abstract

Robert C. Martin, http://www.objectmentor.com/resources/articles/dip.pdf

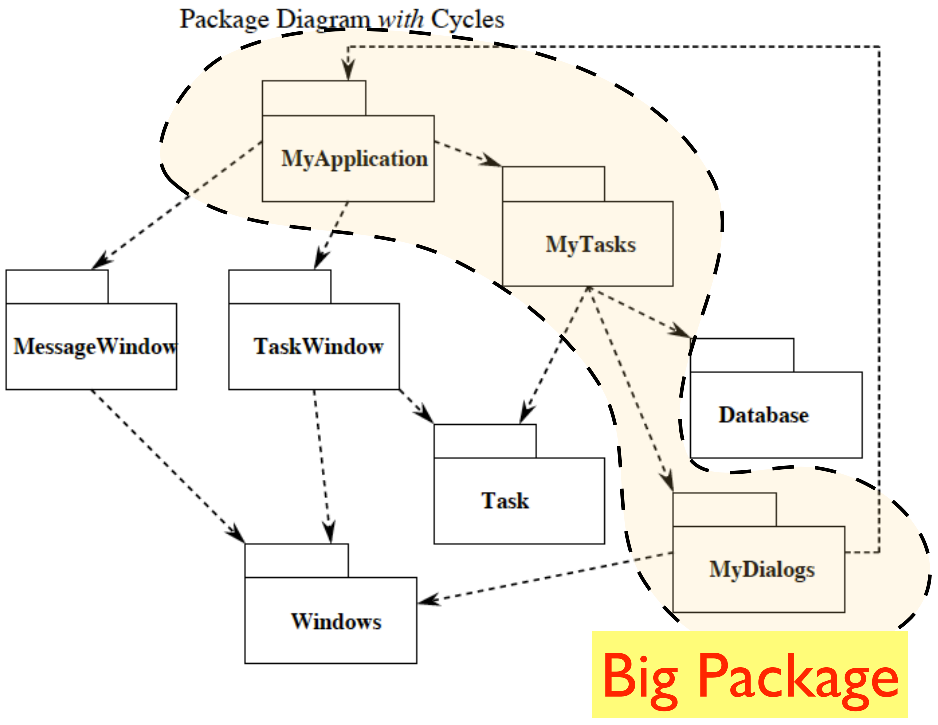
Which is Changeable?
Copy or Device





- *The Reuse/Release Equivalence Principle (REP).*
  - Reuse unit should be used for release unit
    - Reuse means one can use without modifying or verifying module's internal details like static or dynamic library
    - Release process is for notifying reusable module's changes and supporting replacement with new released module
  - Well-defined package can be used for release and reuse unit
- *The Common Reuse Principle (CRP)*
  - Classes in a package could be reused together
  - Answer for what classes are included in same package
- *The Common Closure Principle (CCP)*
  - Changes could be localized in a package. Classes that are changed together always should be included in same package
  - For a small change, one should not investigate whole package structure

- ***The Acyclic Dependencies Principle (ADP)***
  - Package dependency cycles should not exist
  - the morning after syndrome: Code that verified right in last night is broken in tomorrow moring



To release MyTasks package, one should know and verify whole packages

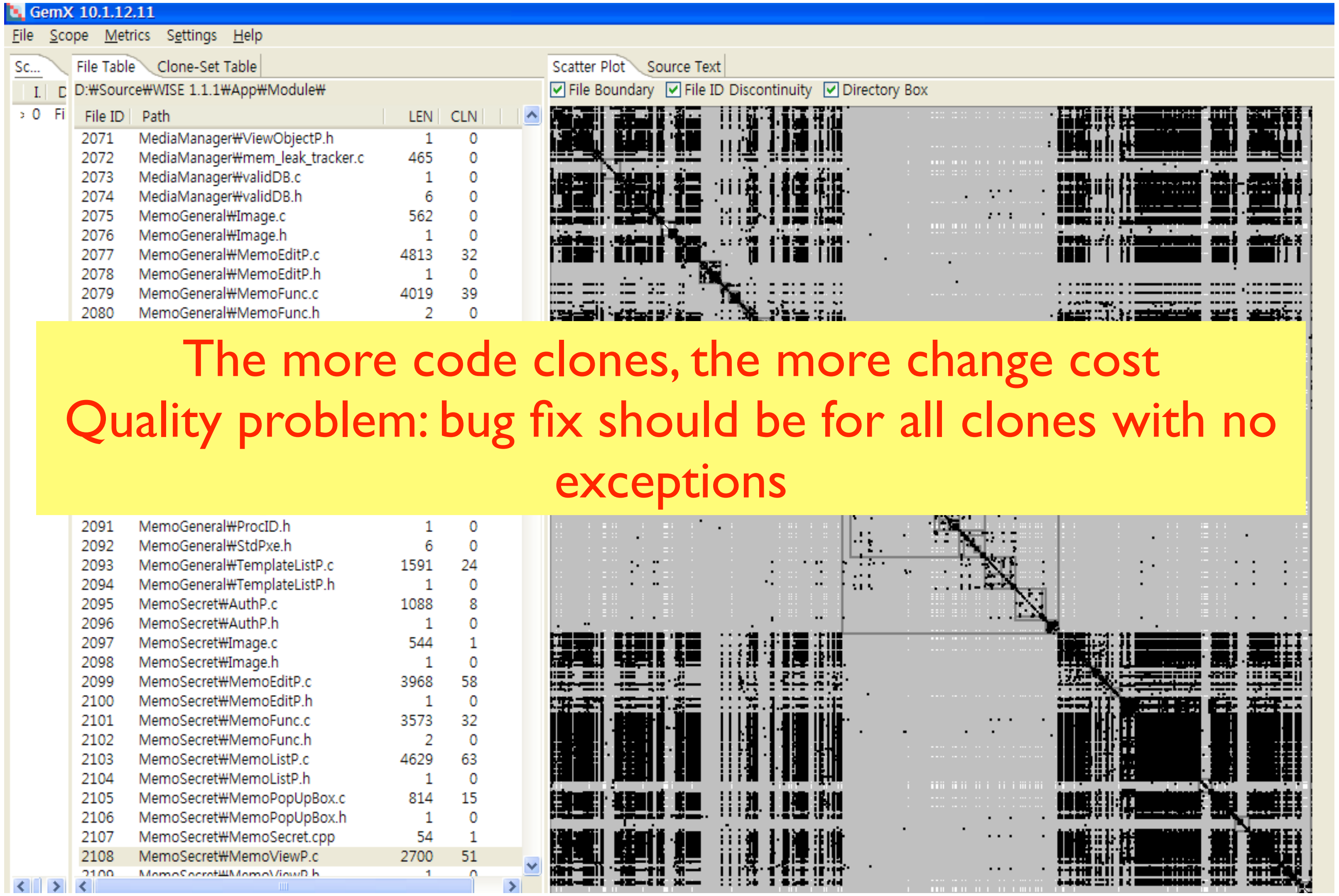
How to break cycles?

Bad smells & Refactoring

- *Bad smells or anti-patterns from "Refactoring by M. Fowler"*
  - **Duplicated code**, Long method, Large class, Long parameter list, **Divergent change**, Shotgun surgery (similar code modification), Feature Envy...
- *Excessive Code Changes at the late phase of development*
  - Mobile phone & TV: On average, 70%+ defects are related with software, 6+ times test cycle

Code clone analysis for 5669 application files

Feature Phone Code with CCFinder

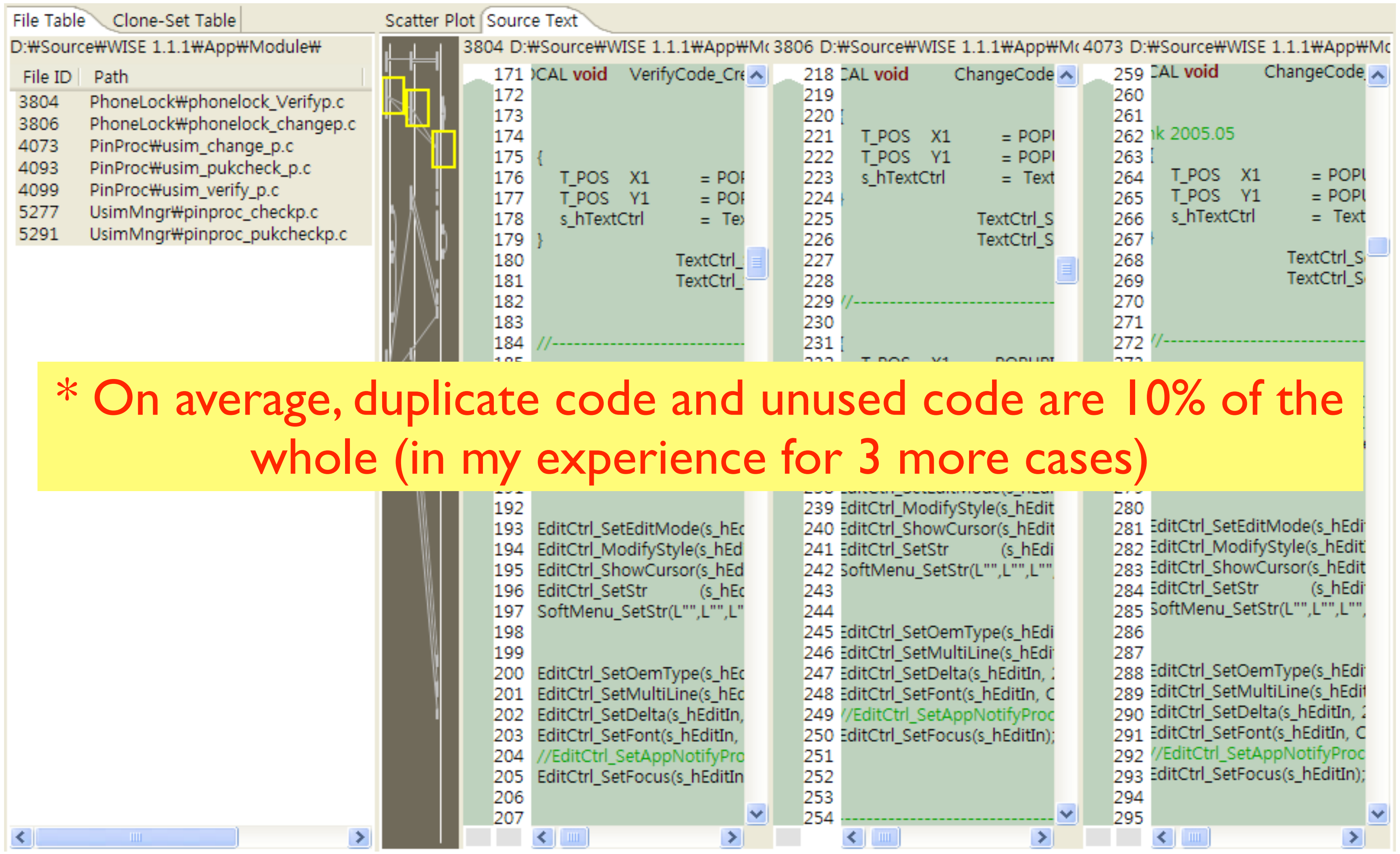

GemX 10.1.12.11
File Scope Metrics Settings Help
Sc... File Table Clone-Set Table
Scatter Plot Source Text
File Boundary File ID Discontinuity Directory Box

D:₩Source₩WISE 1.1.1₩App₩Module₩

File ID	Path	LEN	CLN
2071	MediaManager₩ViewObjectP.h	1	0
2072	MediaManager₩mem_leak_tracker.c	465	0
2073	MediaManager₩validDB.c	1	0
2074	MediaManager₩validDB.h	6	0
2075	MemoGeneral₩Image.c	562	0
2076	MemoGeneral₩Image.h	1	0
2077	MemoGeneral₩MemoEditP.c	4813	32
2078	MemoGeneral₩MemoEditP.h	1	0
2079	MemoGeneral₩MemoFunc.c	4019	39
2080	MemoGeneral₩MemoFunc.h	2	0
2091	MemoGeneral₩ProcID.h	1	0
2092	MemoGeneral₩StdPxe.h	6	0
2093	MemoGeneral₩TemplateListP.c	1591	24
2094	MemoGeneral₩TemplateListP.h	1	0
2095	MemoSecret₩AuthP.c	1088	8
2096	MemoSecret₩AuthP.h	1	0
2097	MemoSecret₩Image.c	544	1
2098	MemoSecret₩Image.h	1	0
2099	MemoSecret₩MemoEditP.c	3968	58
2100	MemoSecret₩MemoEditP.h	1	0
2101	MemoSecret₩MemoFunc.c	3573	32
2102	MemoSecret₩MemoFunc.h	2	0
2103	MemoSecret₩MemoListP.c	4629	63
2104	MemoSecret₩MemoListP.h	1	0
2105	MemoSecret₩MemoPopUpBox.c	814	15
2106	MemoSecret₩MemoPopUpBox.h	1	0
2107	MemoSecret₩MemoSecret.cpp	54	1
2108	MemoSecret₩MemoViewP.c	2700	51
2109	MemoSecret₩MemoViewP.h	1	0

The more code clones, the more change cost

Quality problem: bug fix should be for all clones with no exceptions



D:₩Source₩WISE 1.1.1₩App₩Module₩

File ID	Path
3804	PhoneLock₩phonelock_Verifyp.c
3806	PhoneLock₩phonelock_changep.c
4073	PinProc₩usim_change_p.c
4093	PinProc₩usim_pukcheck_p.c
4099	PinProc₩usim_verify_p.c
5277	UsimMngr₩pinproc_checkp.c
5291	UsimMngr₩pinproc_pukcheckp.c

3804 D:₩Source₩WISE 1.1.1₩App₩Mc

```
)CAL void    VerifyCode_Cre


{
    T_POS   X1          = POF
    T_POS   Y1          = POF
    s_hTextCtrl         = Tex
}
                  TextCtrl_
                  TextCtrl_


//-----------------------------

EditCtrl_SetEditMode(s_hEc
EditCtrl_ModifyStyle(s_hEd
EditCtrl_ShowCursor(s_hEd
EditCtrl_SetStr          (s_hEc
SoftMenu_SetStr(L"",L"",L"

EditCtrl_SetOemType(s_hEc
EditCtrl_SetMultiLine(s_hEc
EditCtrl_SetDelta(s_hEditIn,
EditCtrl_SetFont(s_hEditIn,
//EditCtrl_SetAppNotifyPro
EditCtrl_SetFocus(s_hEditIn
```

3806 D:₩Source₩WISE 1.1.1₩App₩Mc

```
CAL void         ChangeCode

[
  T_POS   X1          = POPI
  T_POS   Y1          = POPI
  s_hTextCtrl         =  Text
}
                  TextCtrl_S
                  TextCtrl_S


//---------------------------

EditCtrl_ModifyStyle(s_hEdit
EditCtrl_ShowCursor(s_hEdit
EditCtrl_SetStr          (s_hEdi
SoftMenu_SetStr(L"",L"",L""

EditCtrl_SetOemType(s_hEdi
EditCtrl_SetMultiLine(s_hEdi
EditCtrl_SetDelta(s_hEditIn, :
EditCtrl_SetFont(s_hEditIn, C
//EditCtrl_SetAppNotifyProc
EditCtrl_SetFocus(s_hEditIn);



---------------------------
```

4073 D:₩Source₩WISE 1.1.1₩App₩Mc

```
CAL void         ChangeCode


ik 2005.05
[
  T_POS   X1          = POPU
  T_POS   Y1          = POPU
  s_hTextCtrl         =  Text
}
                  TextCtrl_S
                  TextCtrl_S


//---------------------------

EditCtrl_SetEditMode(s_hEdi
EditCtrl_ModifyStyle(s_hEdit
EditCtrl_ShowCursor(s_hEdit
EditCtrl_SetStr          (s_hEdi
SoftMenu_SetStr(L"",L"",L"",

EditCtrl_SetOemType(s_hEdi
EditCtrl_SetMultiLine(s_hEdi
EditCtrl_SetDelta(s_hEditIn, :
EditCtrl_SetFont(s_hEditIn, C
//EditCtrl_SetAppNotifyProc
EditCtrl_SetFocus(s_hEditIn);
```

* On average, duplicate code and unused code are 10% of the whole (in my experience for 3 more cases)

Analysis of code changes in derived models

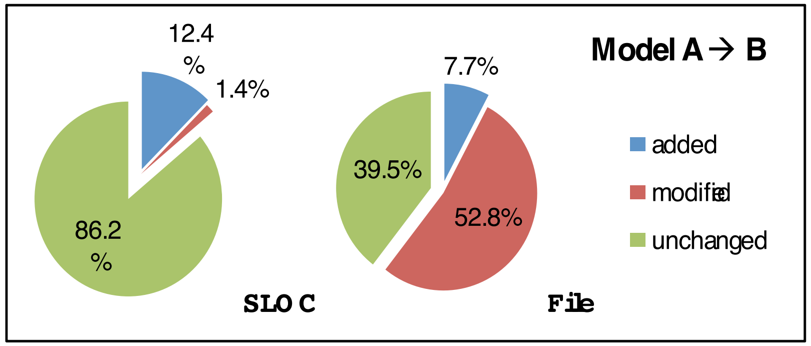
Changed SLOC & The number of changed files

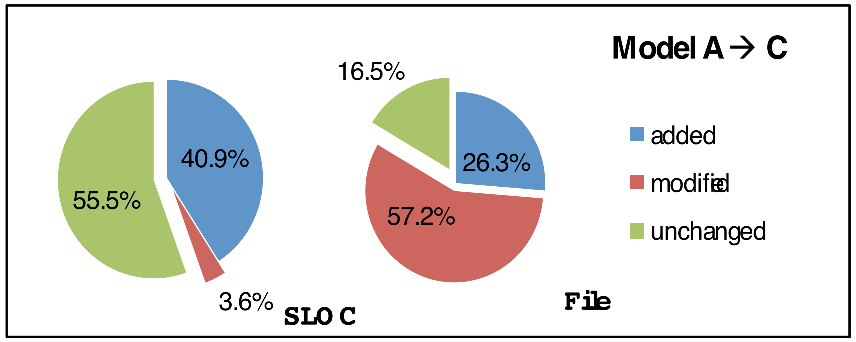
< B case >

Bar type to Slide type
Minor Change

< C case >

Larger LCD, Keypad model to Touch
Major Change



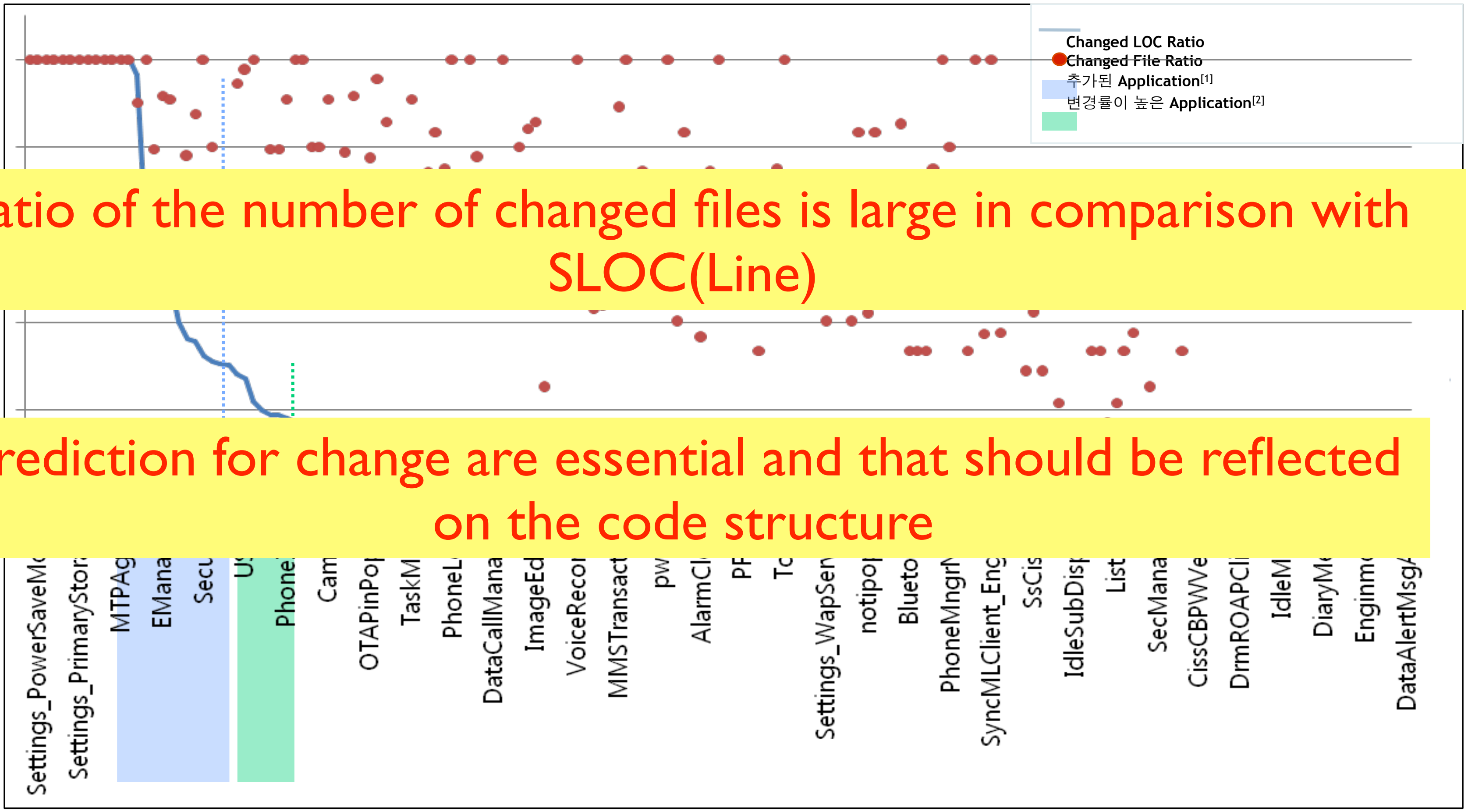


Minor 변경 임에도 불구하고, 전체 파일 수의 반 이상이 수정
많은 파일들을 살펴봐야 하므로, 작업 **Effort**가 커짐
(**Delocalized Changes**)

Major 변경의 경우, **80%**이상의 파일에서 추가/변경이 일어나
며 **45%**의 소스 코드가 수정됨

Divergent or Scattered Changes

In B Case, Code change ratio for each application directory



Ratio of the number of changed files is large in comparison with SLOC(Line)

Prediction for change are essential and that should be reflected on the code structure

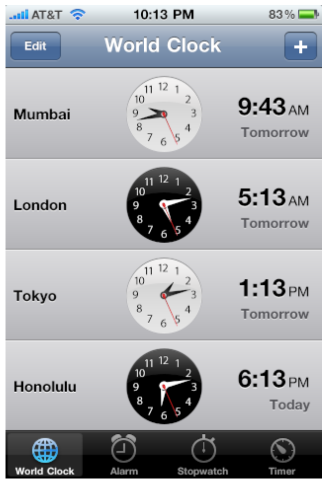
코드 재구조화

- *Flat architecture in World Clock*

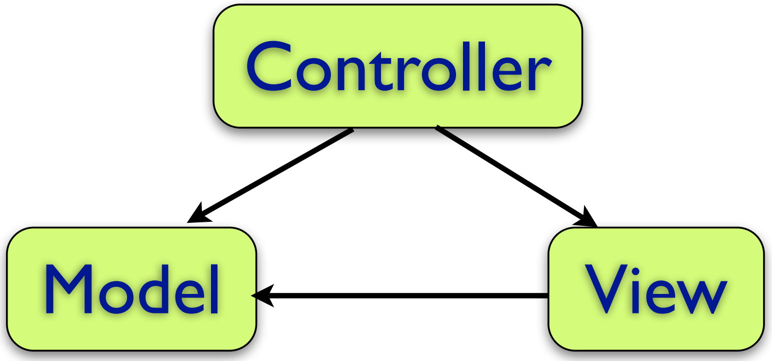
All in a basket?

```
ChangeCity (int)
SetCurrentHomeID (int)
GetCurrentHomeID (void)
DrawHvLine (void)
DrawTimePane (void)
DrawSubTitle (void)
OnInit (void)
OnExit (void)
OnAwake (void)
OnKeyDown (KEY Key)
....
```

Changes



Processing Event by User

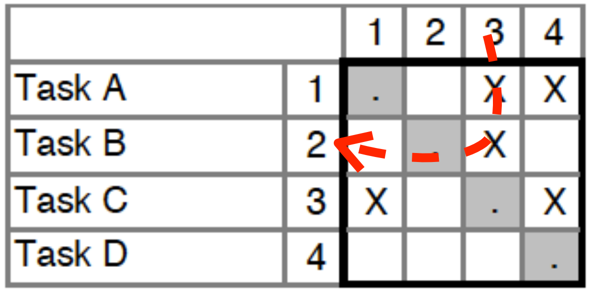


Managine city's time information

Drawing

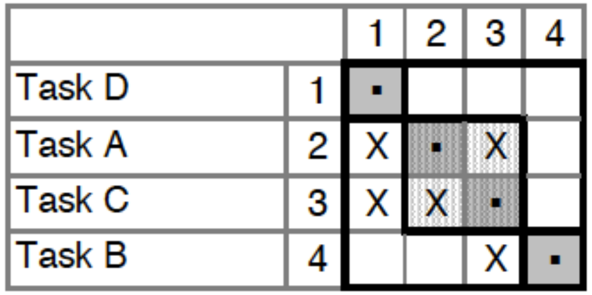
Two commercial tools for structure analysis

Lattix



		1	2	3	4
Task A	1	.		X	X
Task B	2		.	X	
Task C	3	X		.	X
Task D	4				.

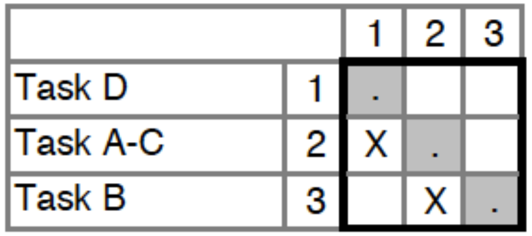
Figure 1: A Simple DSM



		1	2	3	4
Task D	1	▪			
Task A	2	X	▪	X	
Task C	3	X	X	▪	
Task B	4			X	▪

Figure 2: Block Triangular DSM after Partitioning

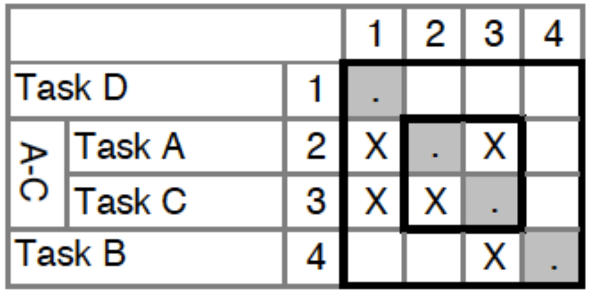
Acyclic = Lower triangular matrix



		1	2	3
Task D	1	.		
Task A-C	2	X	.	
Task B	3		X	.

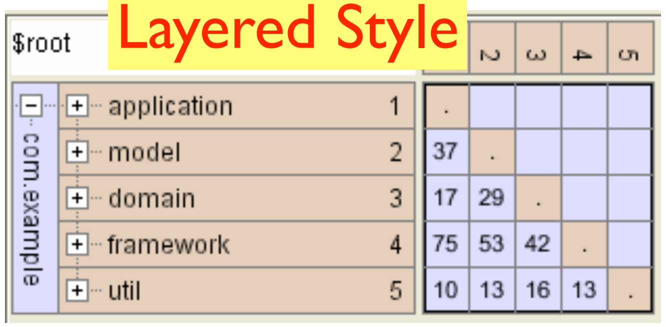
Figure 3: Lower Triangular DSM

Simple cycle elimination by Merging



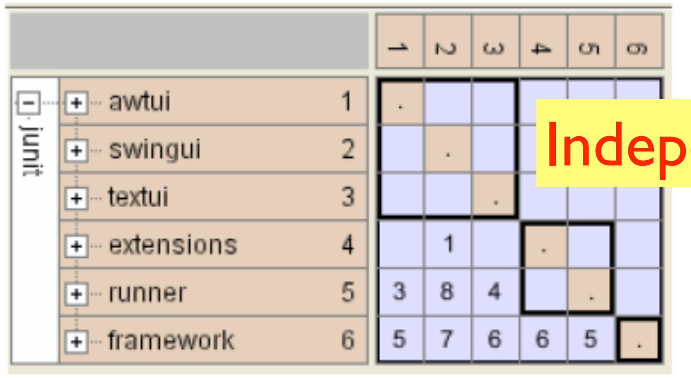
			1	2	3	4
Task D		1	.			
A-C	Task A	2	X	.	X	
	Task C	3	X	X	.	
Task B		4			X	.

Figure 4: Hierarchical DSM



Layered Style

$root				2	3	4	5
com.example	application	1	.				
	model	2	37	.			
	domain	3	17	29	.		
	framework	4	75	53	42	.	
	util	5	10	13	16	13	.

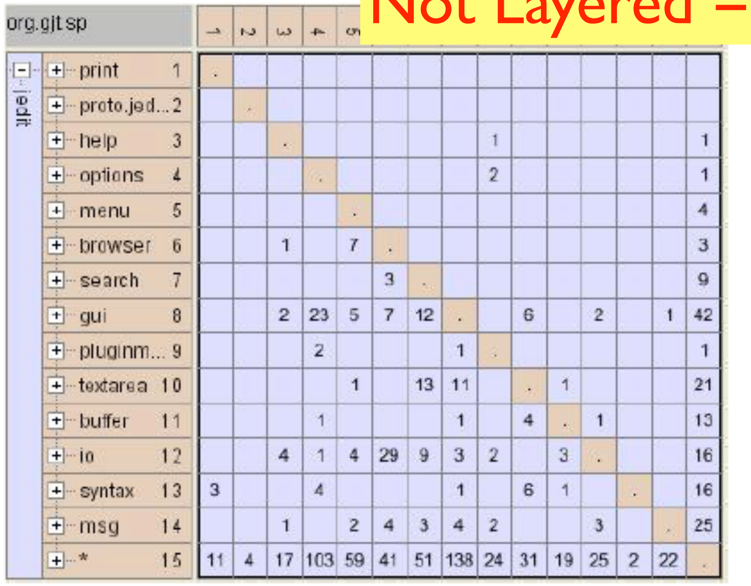


Independent

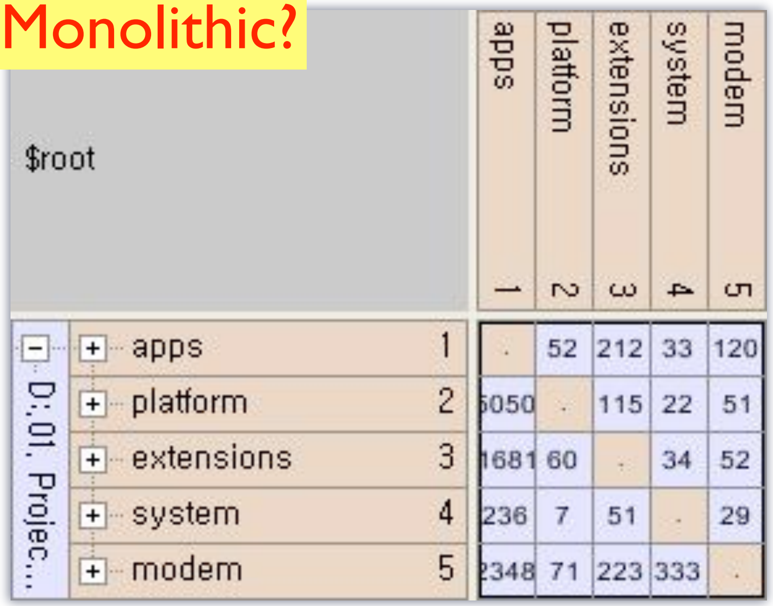
			1	2	3	4	5	6
junit	awtui	1	.					
	swingui	2		.				
	textui	3			.			
	extensions	4		1		.		
	runner	5	3	8	4		.	
	framework	6	5	7	6	6	5	.

Figure 10: DSM for JUnit

Not Layered => Monolithic?



org.gjt.sp			1	2	3	4	5										
jedit	print	1	.														
	proto.jed...	2		.													
	help	3			.						1						1
	options	4				.					2						1
	menu	5					.										4
	browser	6			1		7	.									3
	search	7						3	.								9
	gui	8			2	23	5	7	12	.		6		2		1	42
	pluginm...	9				2				1	.						1
	textarea	10					1		13	11		.	1				21
	buffer	11				1				1		4	.	1			13
	io	12			4	1	4	29	9	3	2		3	.			16
	syntax	13	3			4				1		6	1		.		16
	msg	14			1		2	4	3	4	2			3		.	25
	*	15	11	4	17	103	59	41	51	138	24	31	19	25	2	22	.



$root			apps 1	platform 2	extensions 3	system 4	modem 5
D:.01. Projec...	apps	1	.	52	212	33	120
	platform	2	5050	.	115	22	51
	extensions	3	1681	60	.	34	52
	system	4	236	7	51	.	29
	modem	5	2348	71	223	333	.

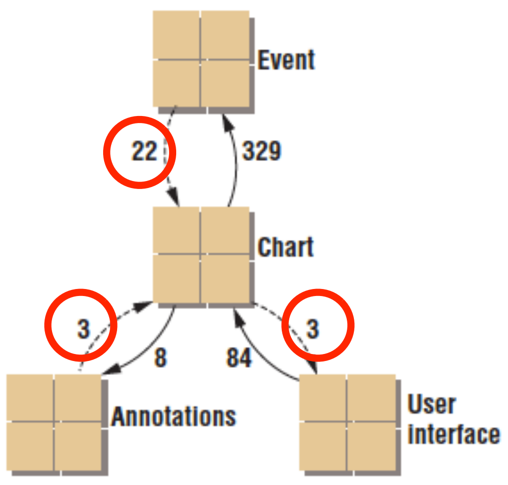
Should investigate every layer violation one by one

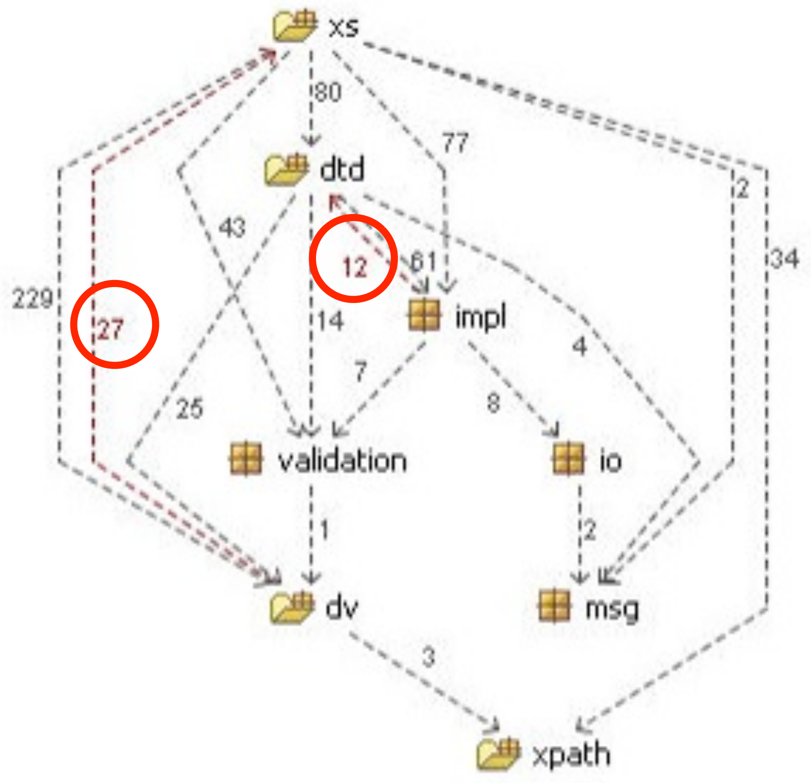
Why modem layer use application layer?

When new application is modified or added, whoever consider modem code in detail?

Structure 101

package-level cycles ("Tangles")
MFS: Minimum Feedback Set of edges to eliminate cycle





org.apache.xerces.impl

Software Engineering Terms

- *Lehman's 1st & 2nd Law of software evolution*
  - Continuing Change
    - A program must be continually adapted or they become progressively less satisfactory
  - Increasing Complexity
    - As a program evolves its complexity increases unless work is done to maintain or reduce it
- *Architecture erosion (D. Perry)*
- *Software Aging (D. L. Parnas)*

Package Layering

Subtype dependency in object-oriented design

Subtype Layers in Object Oriented Framework

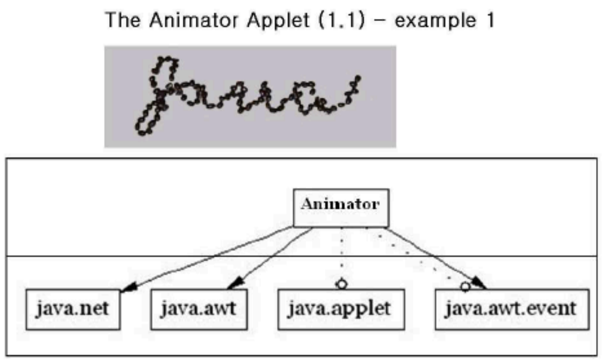
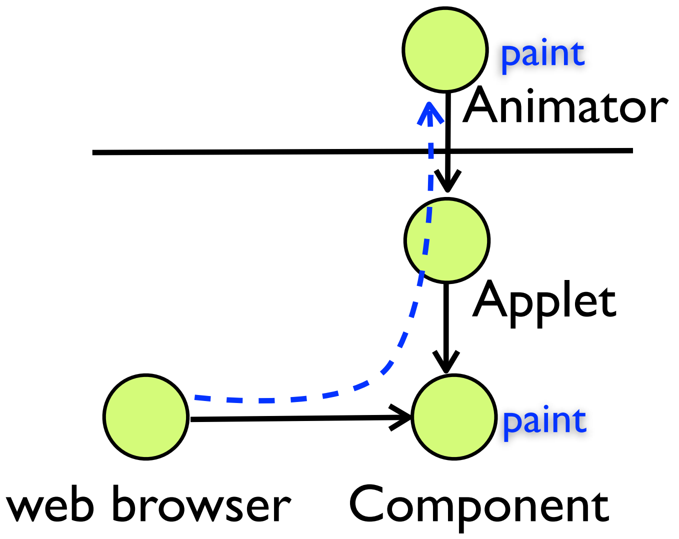
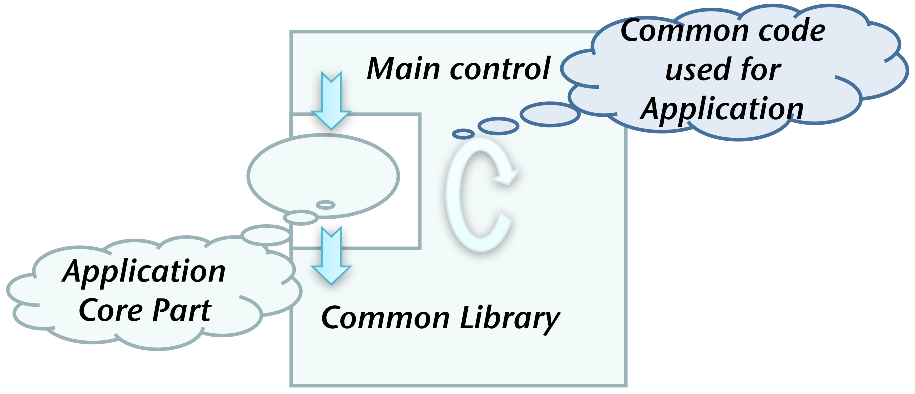


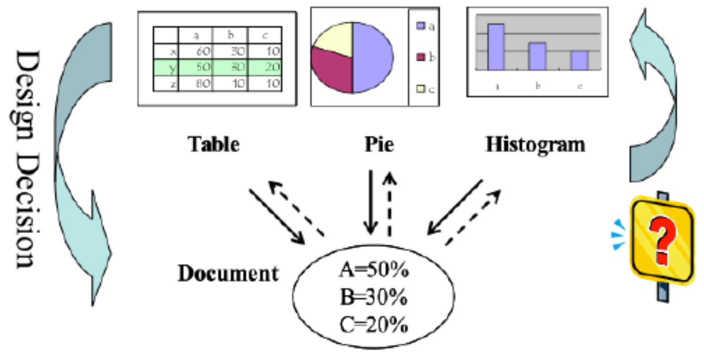
Fig. 1 Typical layers from an application developed with framework classes



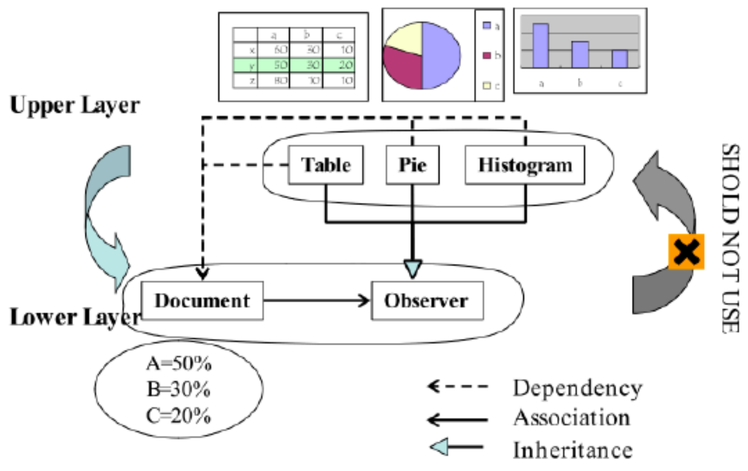
Application Framework



Subtype Layers in Object Oriented Design Patterns



(a) Cyclic dependencies



(b) Layered design by subtype dependency

Bidirectional Communication
=> Bidirectional Dependency?

Poor Reuse, Undesirable Changes

Observer Pattern

Unnamed broadcasting using subtype dependency

Identifying subtype layers

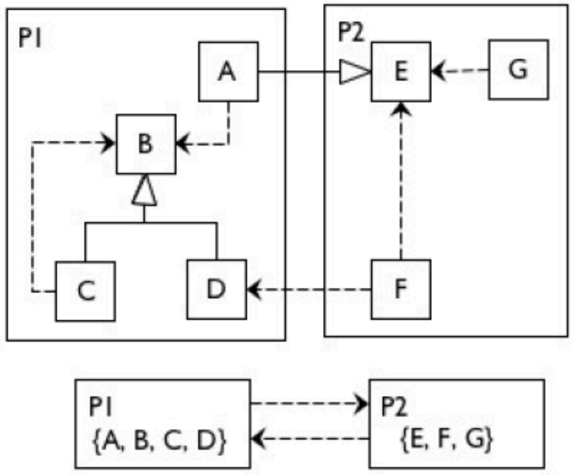


Figure 2: An example of package dependency cycle

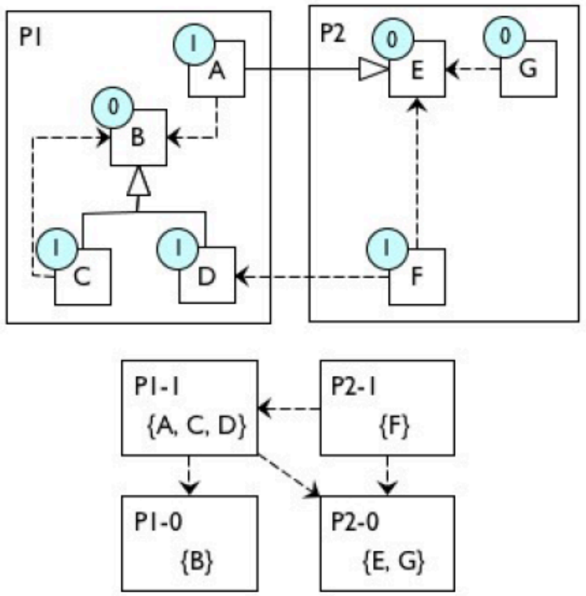


Figure 3: Layered structure for fig. 2

Various types of program dependencies

- Design level (UML)
  Association, Aggregation, Inheritance, Dependency
- Code level
  Import (include), Field type, method type(return, arguments), extends, implements, ...
  Function call, field access, local variable type, ...

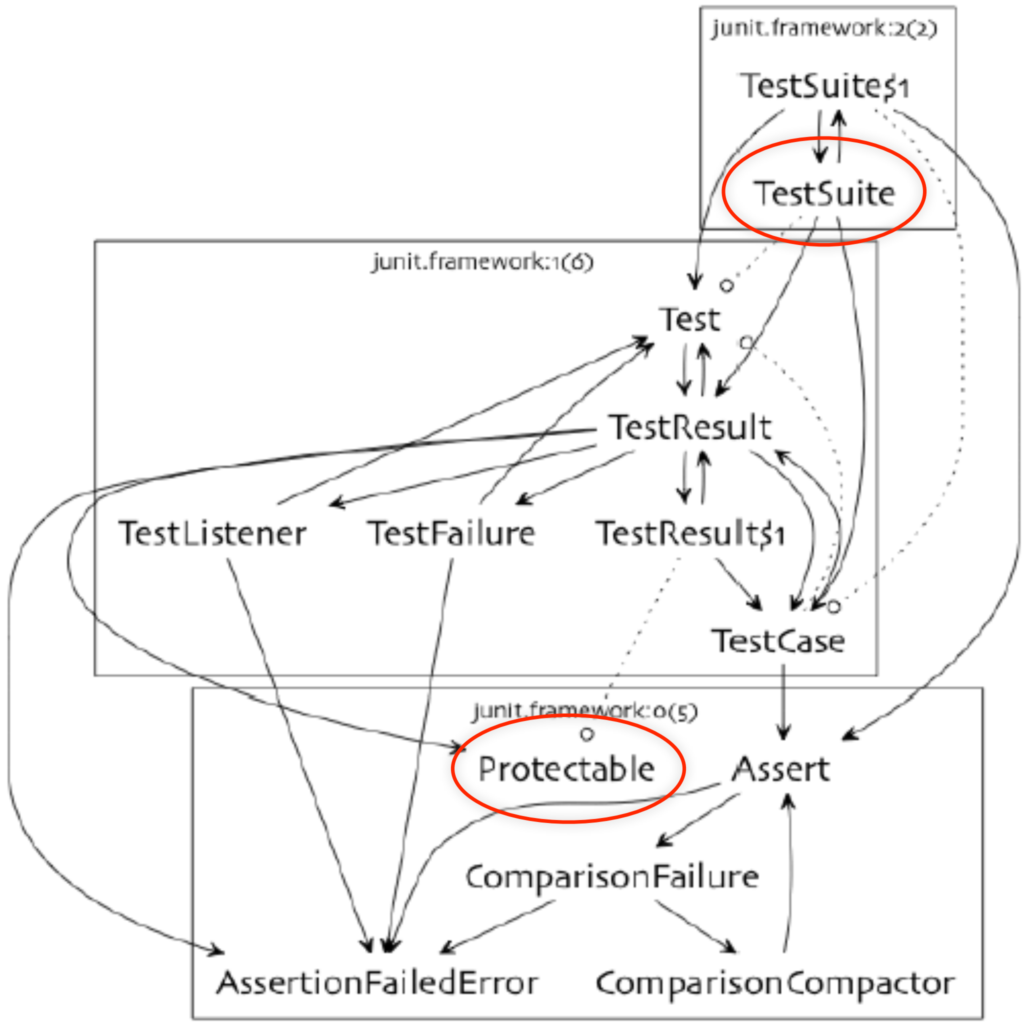
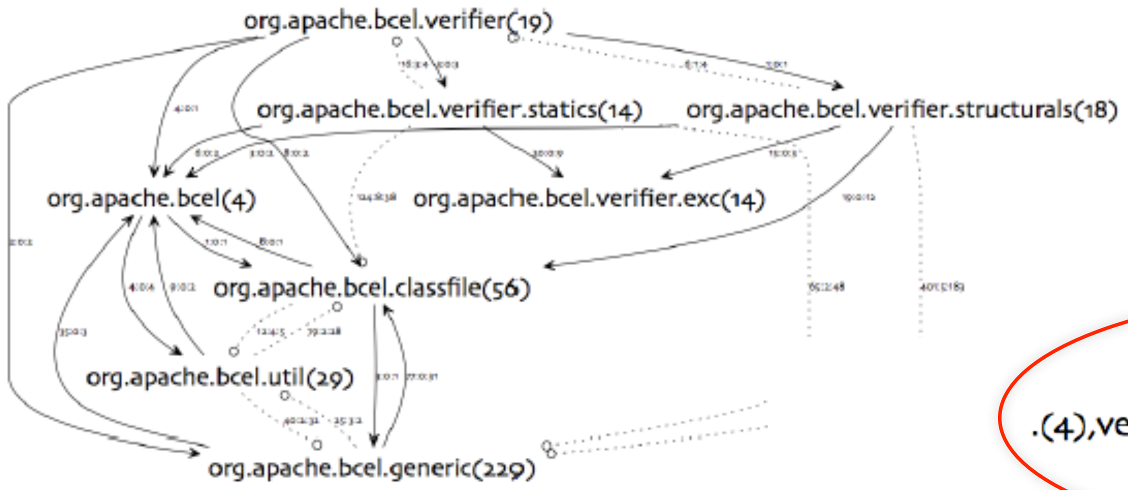


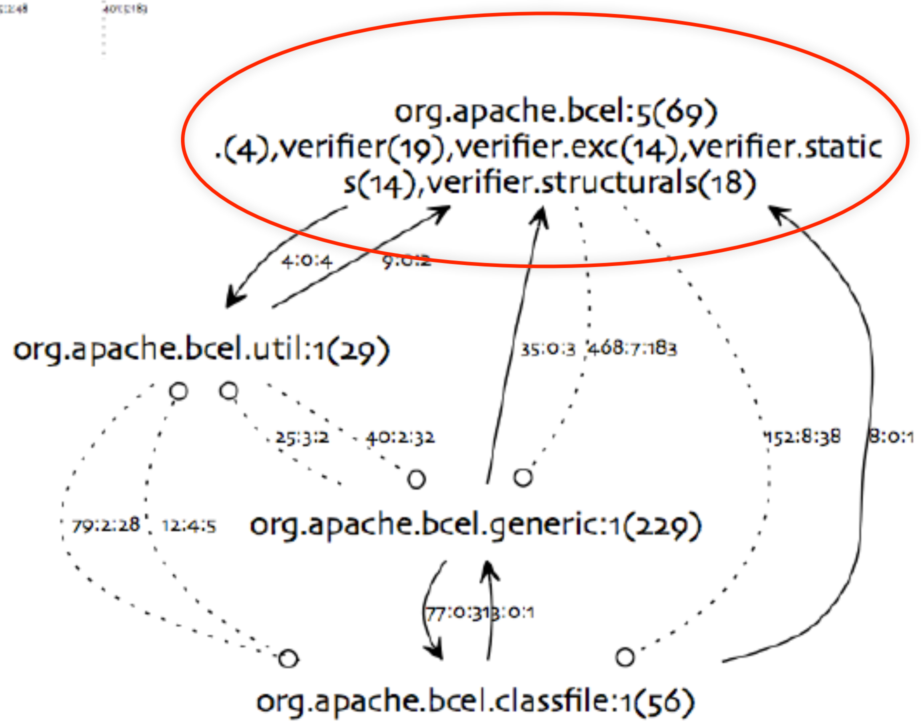
Figure 5: Dependency structure in junit.framework

Simplifying Package Structure



org.apache.bcel.verifier(19)
org.apache.bcel.verifier.statics(14)
org.apache.bcel.verifier.structurals(18)
org.apache.bcel(4)
org.apache.bcel.verifier.exc(14)
org.apache.bcel.classfile(56)
org.apache.bcel.util(29)
org.apache.bcel.generic(229)

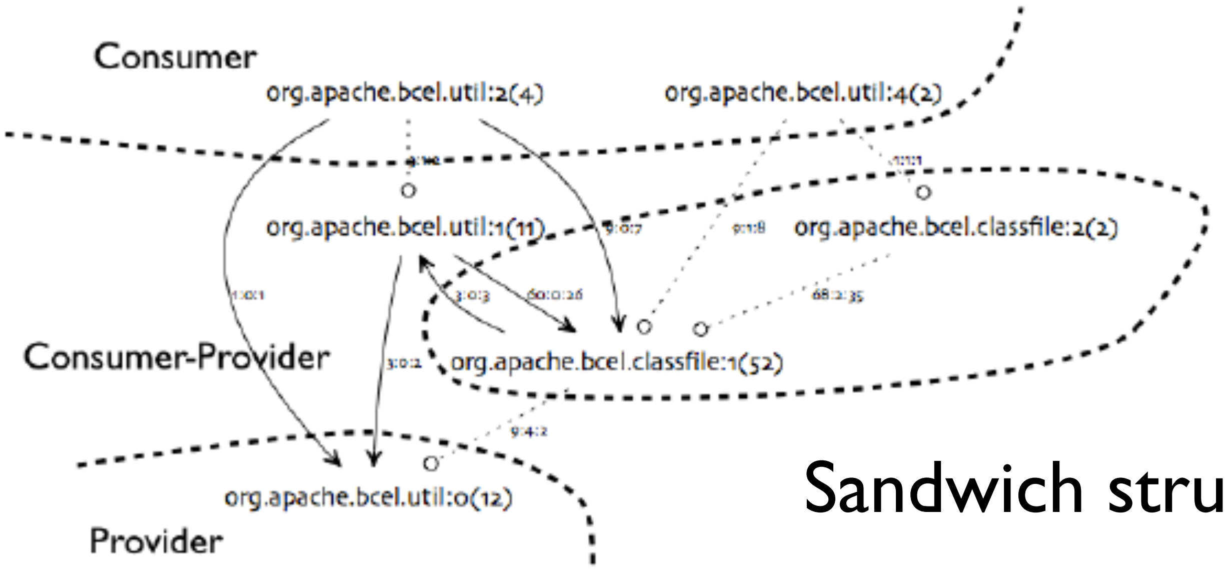
Figure 6: Original package structu



org.apache.bcel:5(69)
.(4),verifier(19),verifier.exc(14),verifier.static
s(14),verifier.structurals(18)
4:0:4
9:0:2
org.apache.bcel.util:1(29)
35:0:3
468:7:183
25:3:2
40:2:32
152:8:38
8:0:1
79:2:28
12:4:5
org.apache.bcel.generic:1(229)
77:0:31
3:0:1
org.apache.bcel.classfile:1(56)

Figure 7: Simplified complex package structure of BCEL

Analyzing package structure with layers

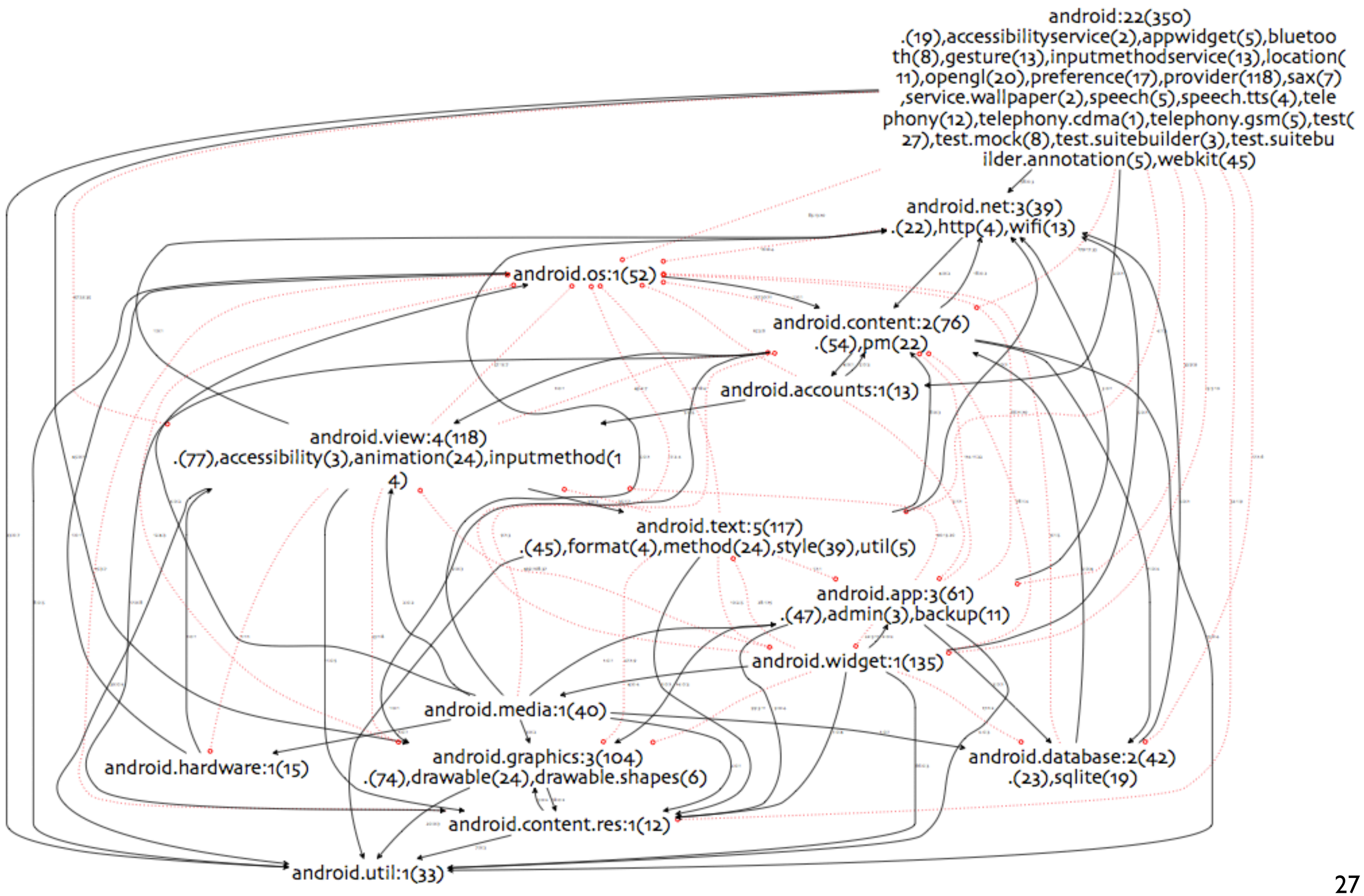

Consumer
org.apache.bcel.util:2(4)
org.apache.bcel.util:4(2)
org.apache.bcel.util:1(11)
org.apache.bcel.classfile:2(2)
Consumer-Provider
org.apache.bcel.classfile:1(52)
org.apache.bcel.util:0(12)
Provider

Sandwich structure

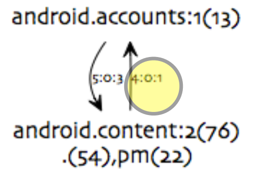
Figure 8: Layered package structured of util and classfile of BCEL

- Util:0 - Internal utilities for other BCEL classes, bytecode comparator, sequence and class path
- Util:1,2,4 - External utilities for non-BCEL classes, HTML generation for Java class

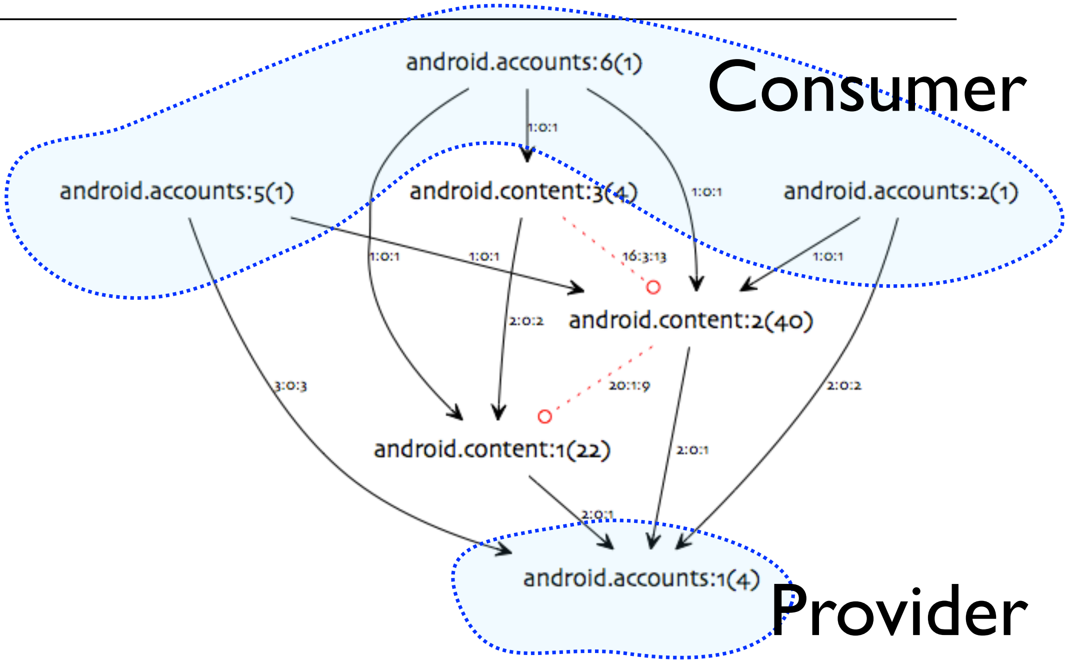
Android’s Package Structure

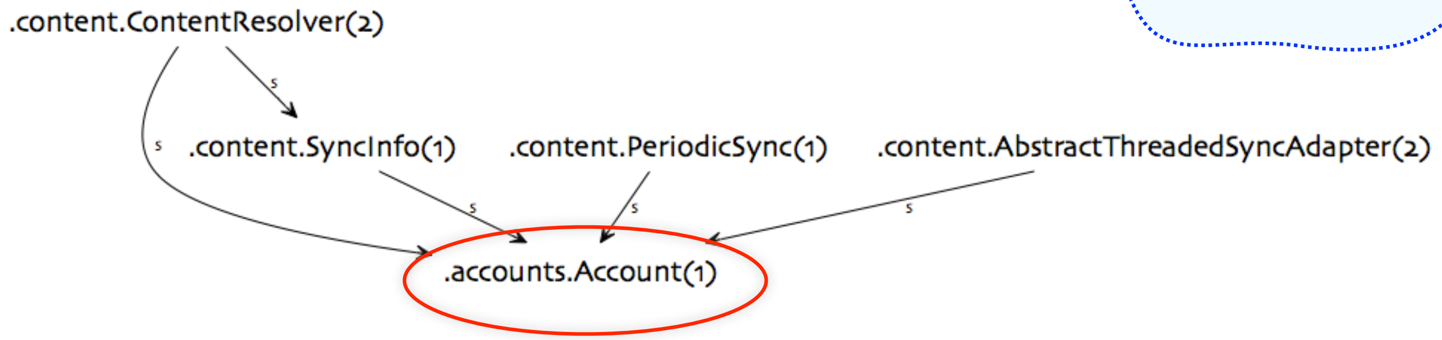

android:22(350) .(19),accessibilityservice(2),appwidget(5),bluetooth(8),gesture(13),inputmethodservice(13),location(11),opengl(20),preference(17),provider(118),sax(7),service.wallpaper(2),speech(5),speech.tts(4),telephony(12),telephony.cdma(1),telephony.gsm(5),test(27),test.mock(8),test.suitebuilder(3),test.suitebuilder.annotation(5),webkit(45)
android.net:3(39) .(22),http(4),wifi(13)
android.os:1(52)
android.content:2(76) .(54),pm(22)
android.accounts:1(13)
android.view:4(118) .(77),accessibility(3),animation(24),inputmethod(14)
android.text:5(117) .(45),format(4),method(24),style(39),util(5)
android.app:3(61) .(47),admin(3),backup(11)
android.widget:1(135)
android.media:1(40)
android.hardware:1(15)
android.graphics:3(104) .(74),drawable(24),drawable.shapes(6)
android.database:2(42) .(23),sqlite(19)
android.content.res:1(12)
android.util:1(33)

Cases - content to account



4:0:1



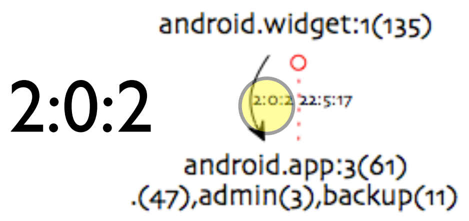


ContentResolver: Provides applications access to the content model
Account is neeed Mostly for synchronizing data(e.g. Calendar)
    public static void removePeriodicSync(Account account, String authority, Bundle extras) {

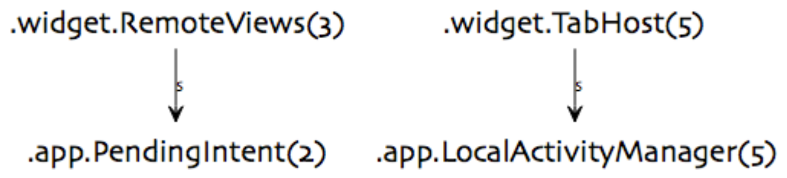
Access to Account could be considered as Ground Rule for android

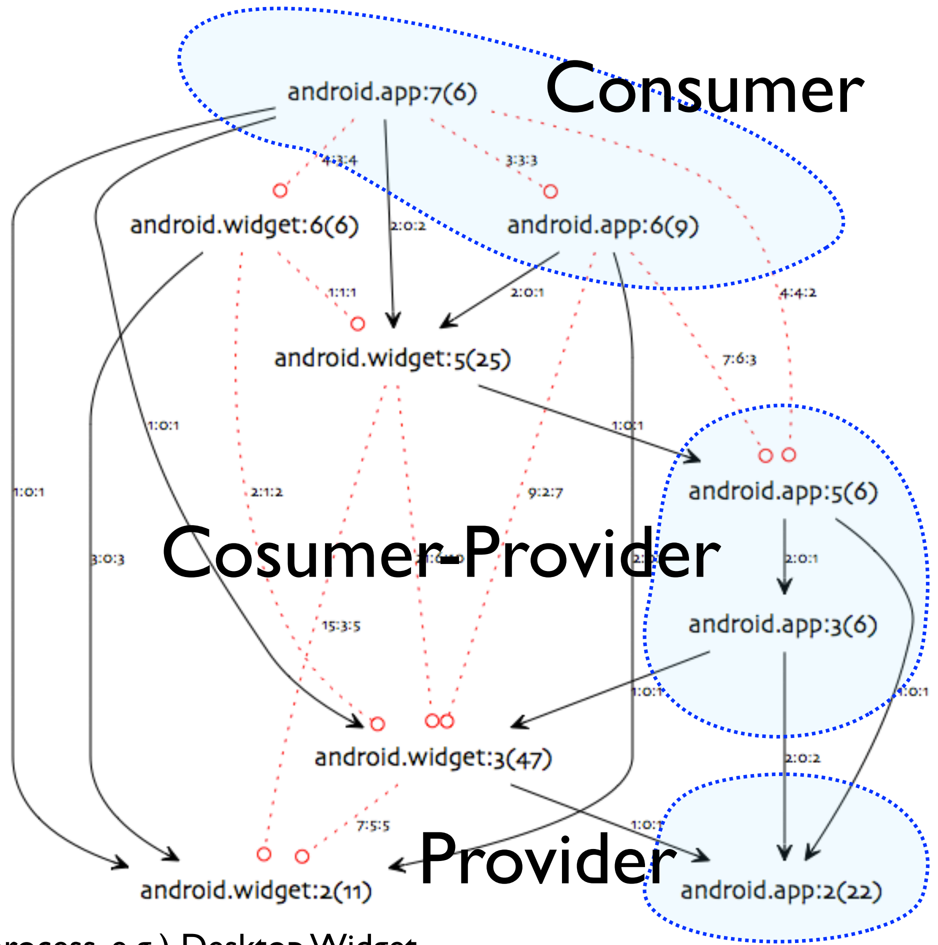
4 Edges

Cases - widget to app



This package (android.app) builds on top of the lower-level Android packages android.widget, android.view, android.content, ... (from JavaDoc file)

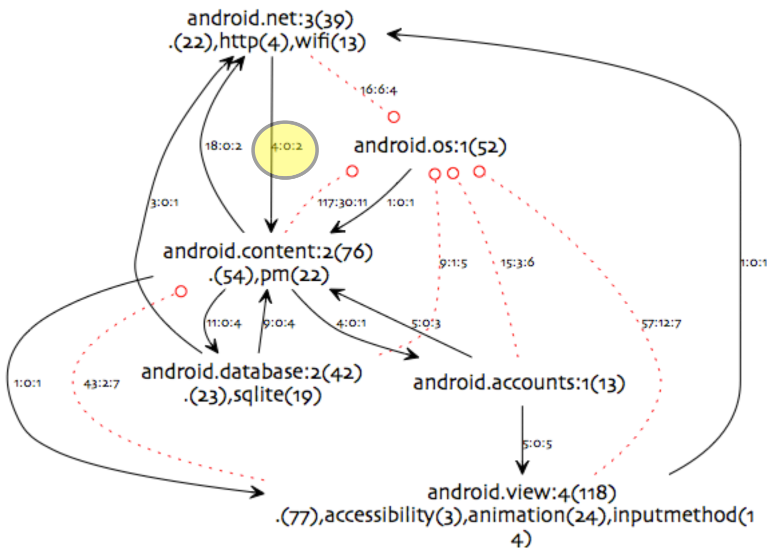


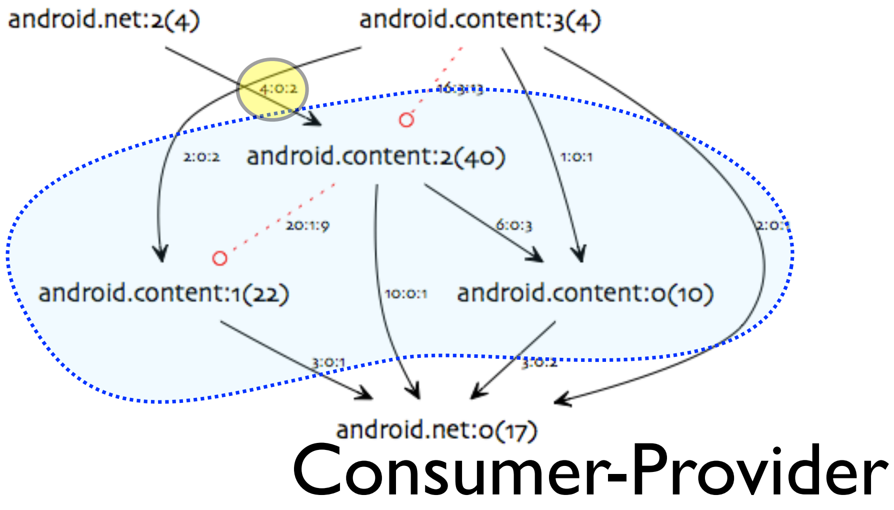


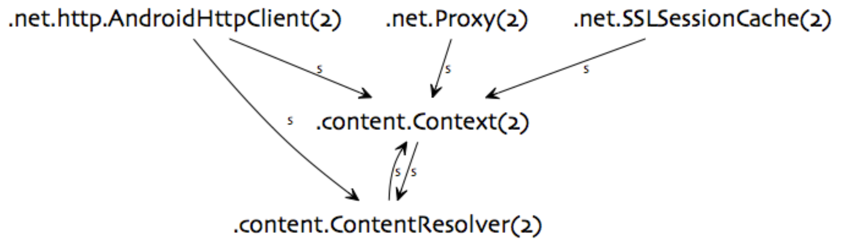
- RemoteViews: A view that can be displayed in another process, e.g.) Desktop Widget
    - A class that describes a view hierarchy that can be displayed in another process.
- PendingIntent
    - By giving a PendingIntent to another application, you are granting it the right to perform the operation you have specified as if the other application was yourself (with the same permissions and identity)

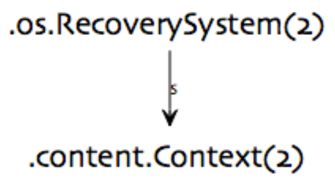
Intent defined in content package is for launching activities
PendingIntent in app package is for obtaining activity information to launch by Desktop Widget. The informaiton is prepared by each activity

Cases - net to content



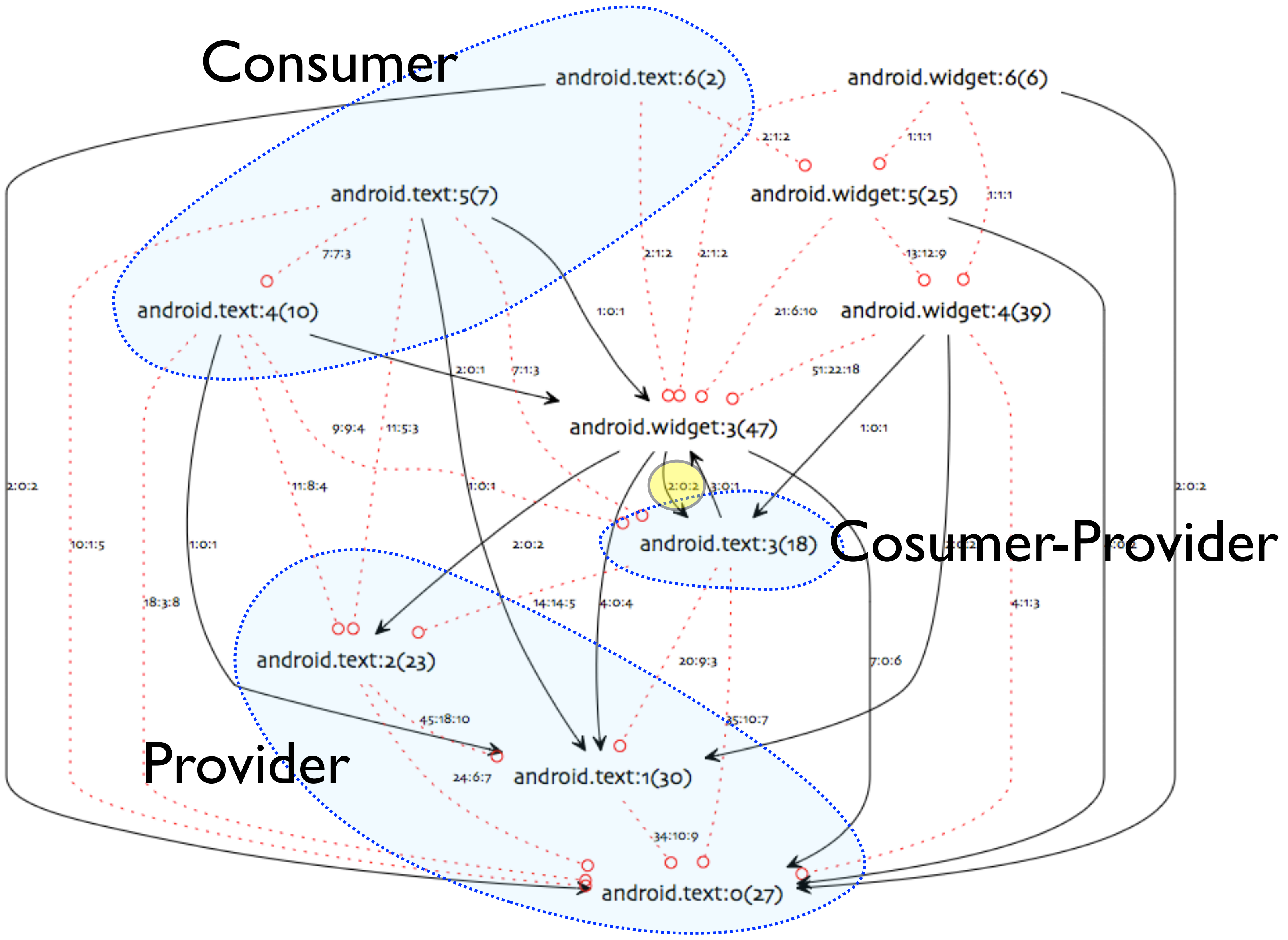




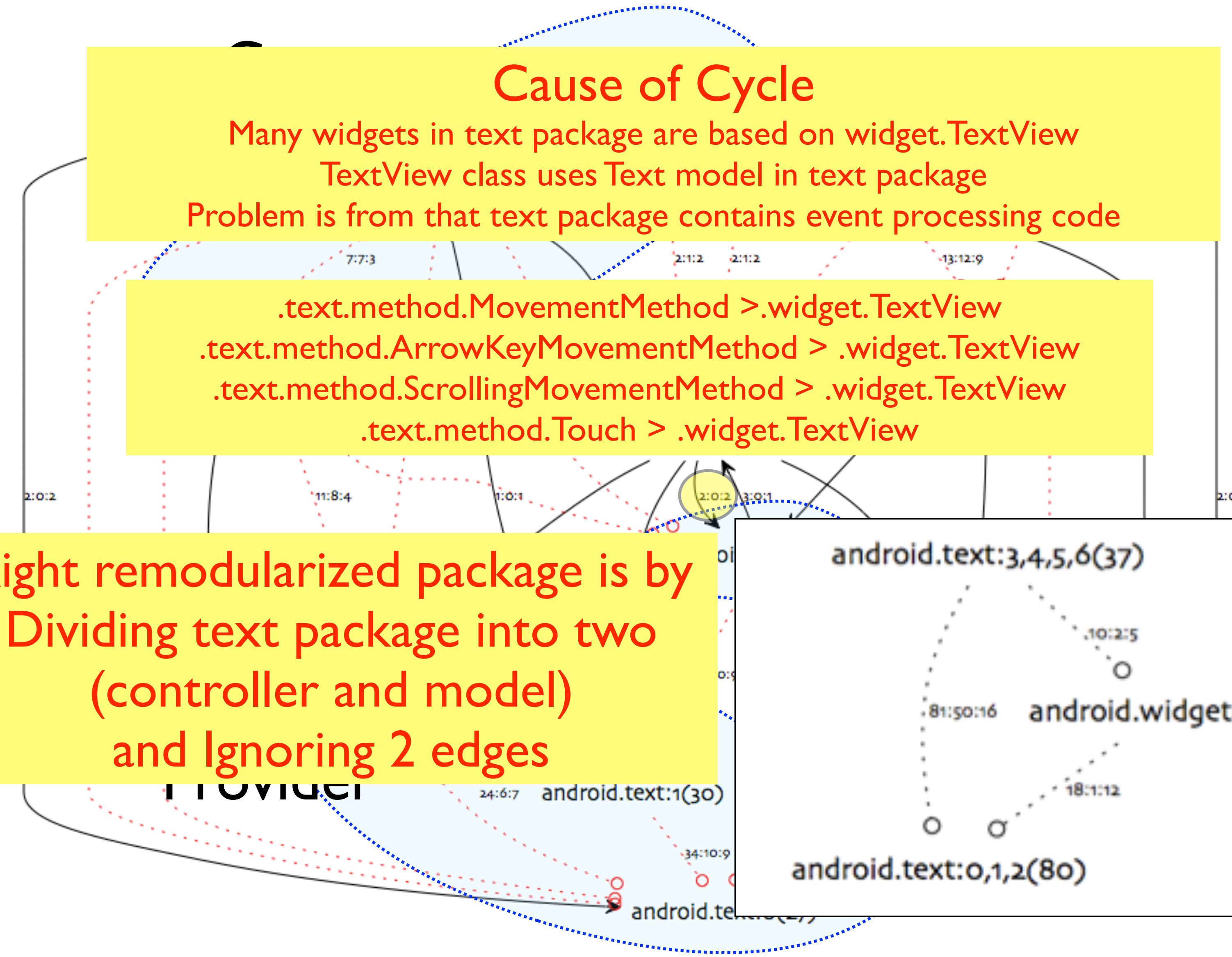


- Context and ContentResolver are used to get directory for storing SessionCache
  - These classes are required to obtain Android system information => Ground Rule
- Context: Interface to global information about an application environment. This is an abstract class whose implementation is provided by the Android system.
  - In SSLSeesionCache, File dir = context.getDir("sslcache", Context.MODE_PRIVATE);
- Dependency from os to content is also for obtaining directory information for system recovery
  - PowerManager pm = (PowerManager) context.getSystemService(Context.POWER_SERVICE);
  - pm.reboot("recovery");

Cases - widget to text



Cases - widget to text

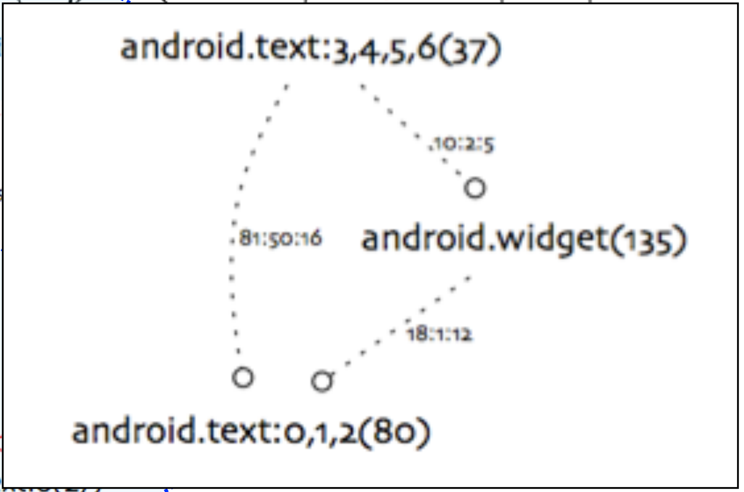


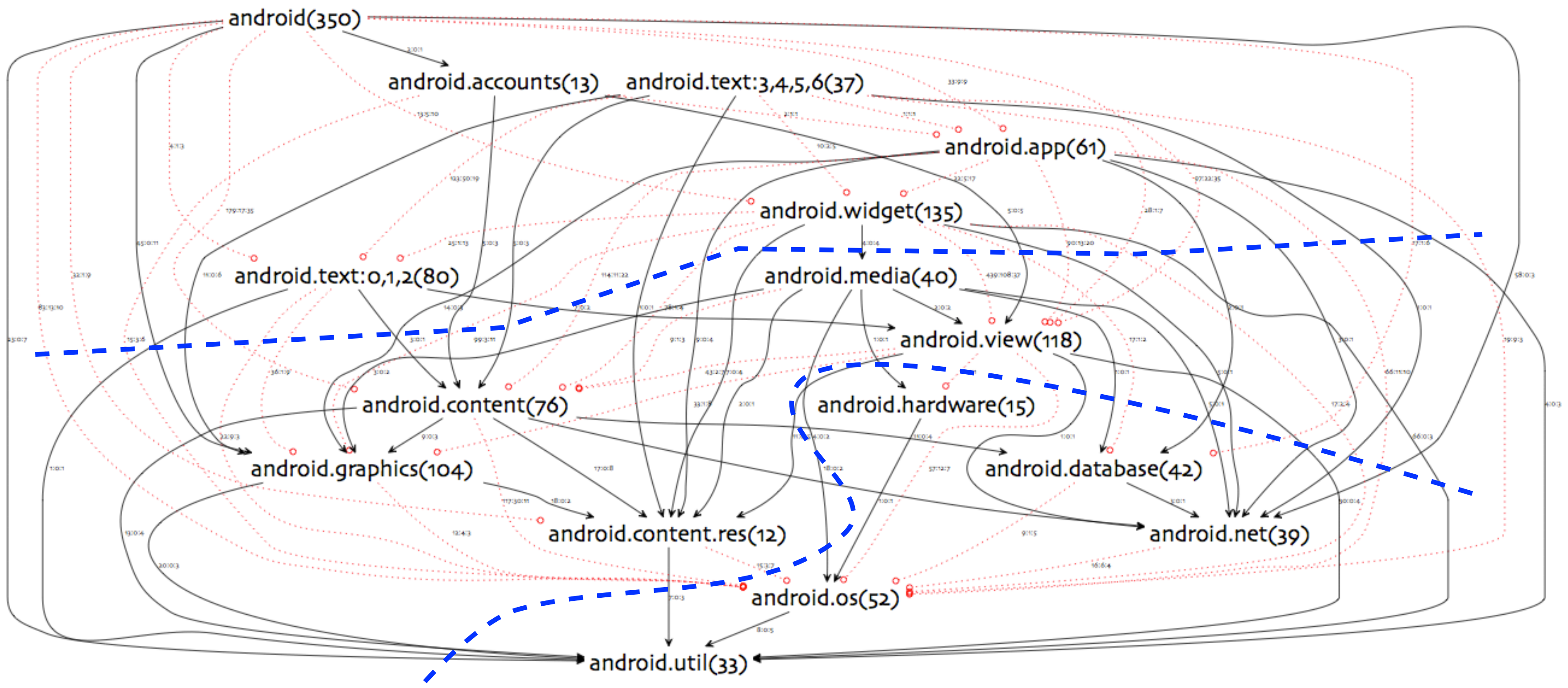
Cause of Cycle

Many widgets in text package are based on widget.TextView
TextView class uses Text model in text package
Problem is from that text package contains event processing code

.text.method.MovementMethod >.widget.TextView
.text.method.ArrowKeyMovementMethod > .widget.TextView
.text.method.ScrollingMovementMethod > .widget.TextView
.text.method.Touch > .widget.TextView

Right remodularized package is by
Dividing text package into two
(controller and model)
and Ignoring 2 edges





Cycle-free Package Structure => Layered Architecture
By Re-modularizing text package & Ignoring 32 edges

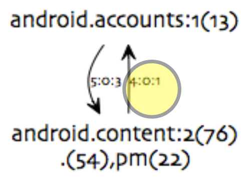
android.media.RingtoneManager>android.app.Activity

Conclusions

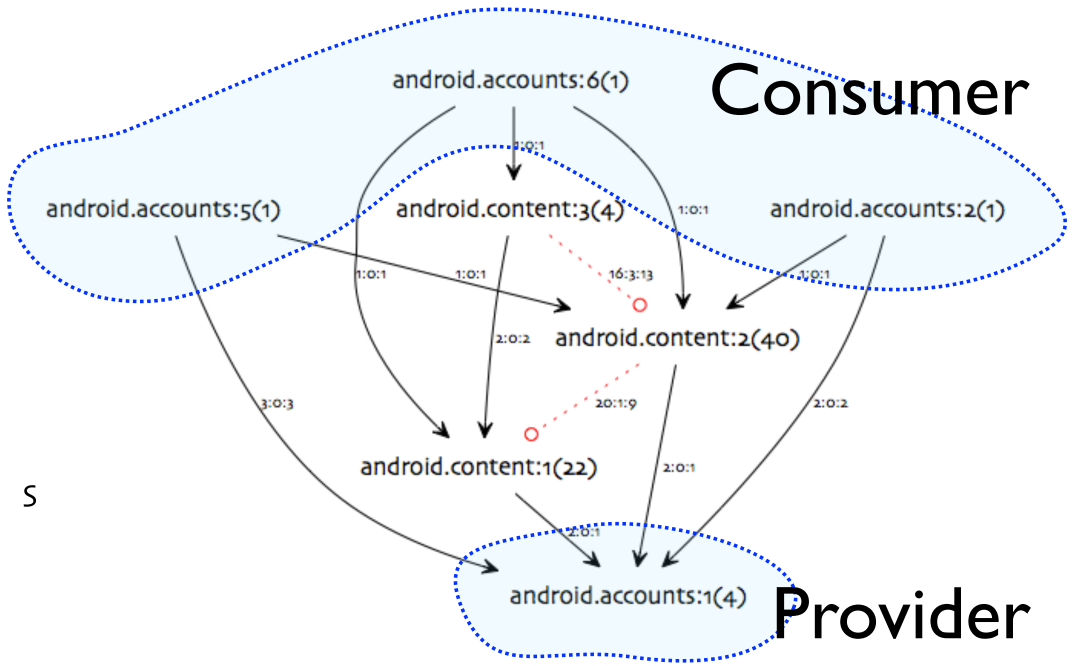
- *Package Structure is important for*
  - Work assignment, daily development, estimation of how much efforts are required for a change request
  - Identification of reusable part, build order, testing order, ...
- *Package structure as an instant architecture*
  - Communication
  - Code structure itself can be used for a design document
  - Helpful when change speed is high and requirements are unstable (architecture agile to changes)
- *At the minimum, package cycles should be managed*
  - Complex package structure is a bad smells that project is not healthy at the current time

Thanks for Listening!!!
Questions?

Cycle 1. accounts



4:0:1

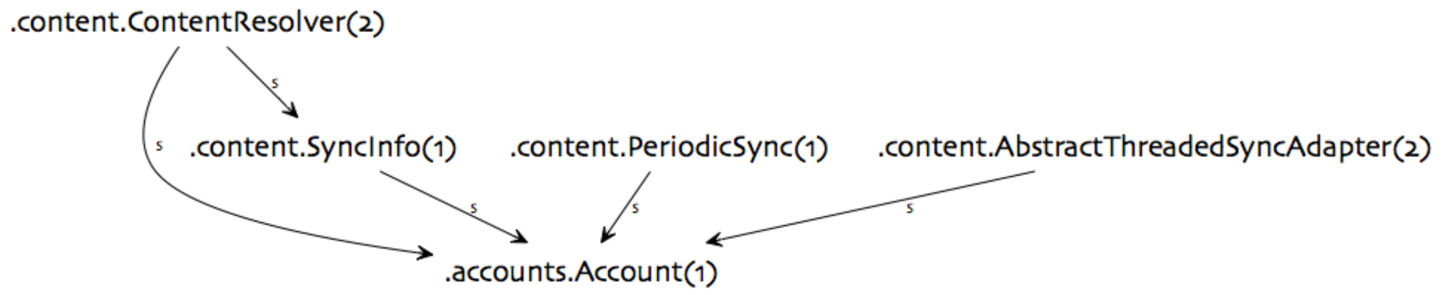


***android.content:1->android.accounts:1

PeriodicSync	**Account**	S
SyncInfo	**Account**	S

***android.content:2->android.accounts:1

AbstractThreadedSyncAdapter	**Account**	S
ContentResolver	**Account**	S



4 Edges

What is problematic edges?

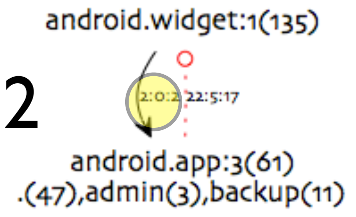
Content to accounts (Less dependencies)

Dependency on Account is ground rule or violation?

Ground rule & Need Documentation

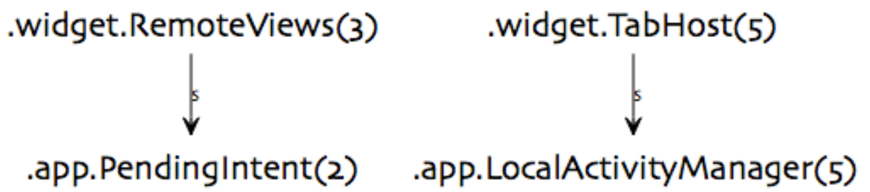
Cycle 2. app

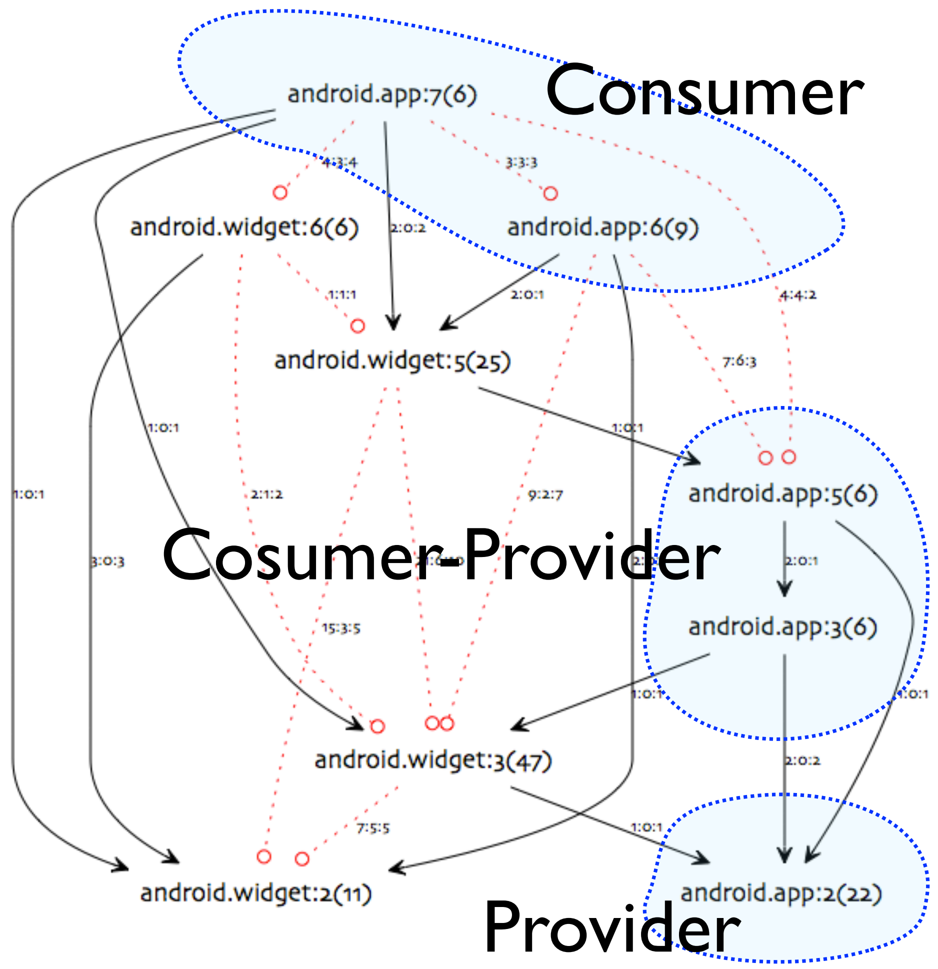
2:0:2



***android.widget:5->android.app:5
TabHost LocalActivityManager S

***android.widget:3->android.app:2
RemoteViews PendingIntent S





2 Edges

What is problematic edges?

widget to app (Less dependencies)

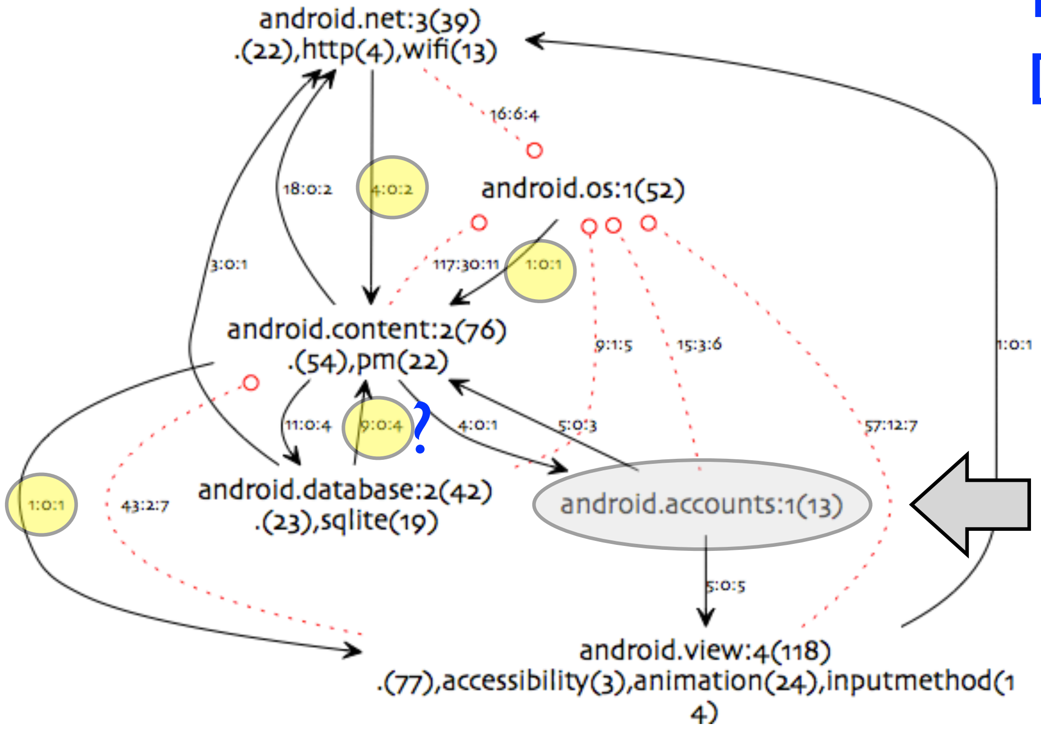
Ground rule or violation?

Violation

For what TabHost uses LocalActivityManager?

For what RemoteViews uses PendingIntent?

Cycle 3. content



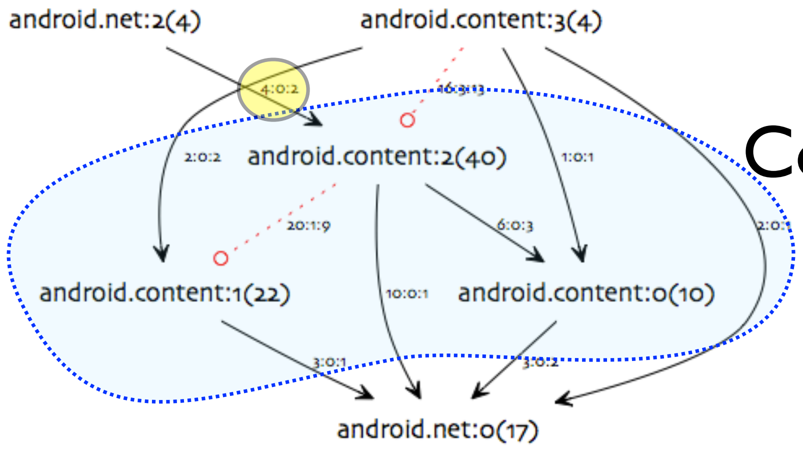
Less dependencies
Design Knowledge

We already solve cycle with accounts

How complex the content's dependencies?
net, os, database, view

Check & Solve One by One !!!

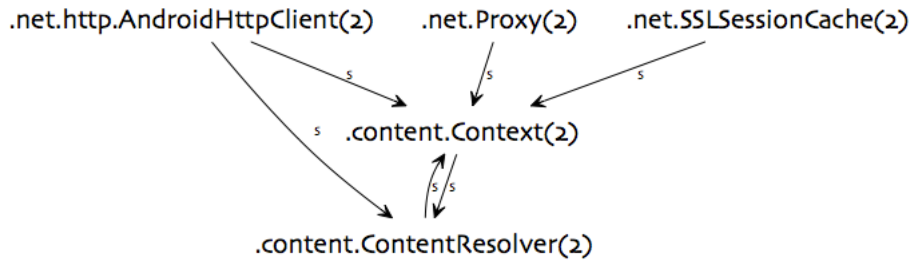
Content - net



Consumer-Provider

***android.net:2->android.content:2

Proxy	Context	S
SSLSessionCache	Context	S
http.AndroidHttpClient	ContentResolver	S
http.AndroidHttpClient	Context	S



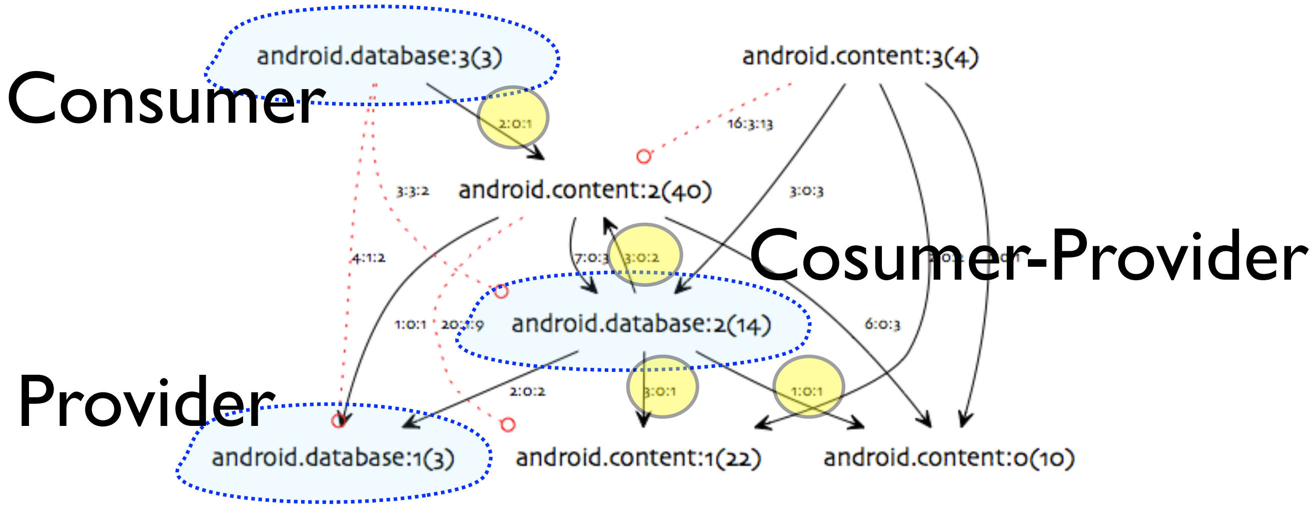
4 Edges

Context, ContentResolver is Special Classes in Android

net to content (Less dependencies)

Ground Rule

Content - database

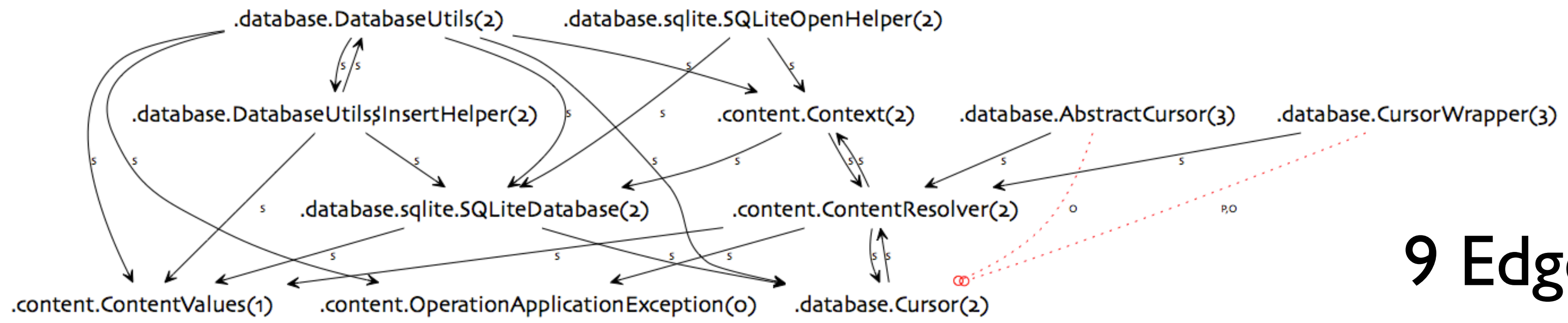


***android.database:3->android.content:2
AbstractCursor ContentResolver S
CursorWrapper ContentResolver S

***android.database:2->android.content:1
DatabaseUtils ContentValues S
DatabaseUtils$InsertHelper ContentValues S
sqlite.SQLiteDatabase ContentValues S

***android.database:2->android.content:2
Cursor ContentResolver S
DatabaseUtils Context S
sqlite.SQLiteOpenHelper Context S

***android.database:2->android.content:0
DatabaseUtils OperationApplicationException S

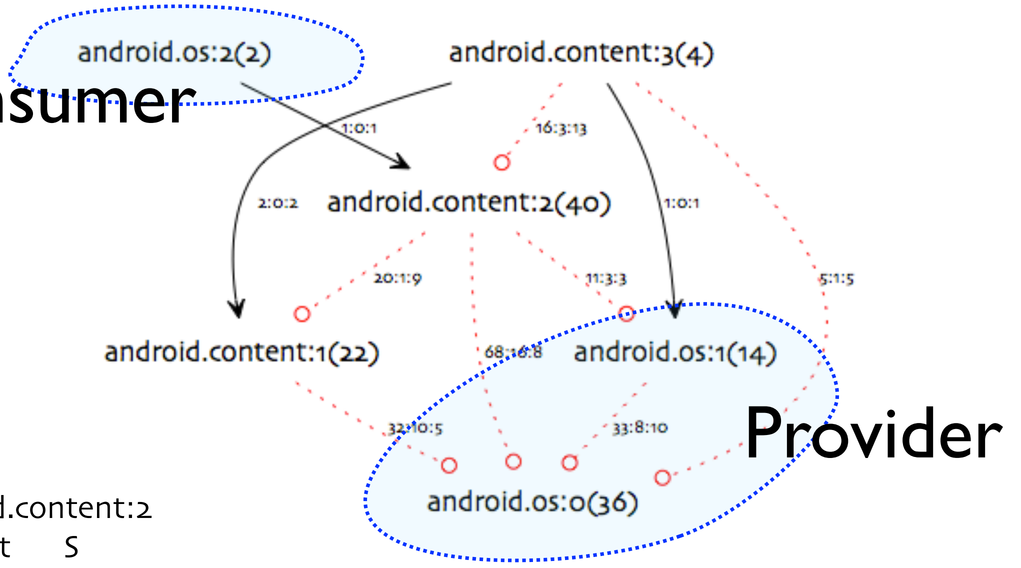

.database.DatabaseUtils(2)
.database.sqlite.SQLiteOpenHelper(2)
.database.DatabaseUtils$InsertHelper(2)
.content.Context(2)
.database.AbstractCursor(3)
.database.CursorWrapper(3)
.database.sqlite.SQLiteDatabase(2)
.content.ContentResolver(2)
.content.ContentValues(1)
.content.OperationApplicationException(0)
.database.Cursor(2)

9 Edges

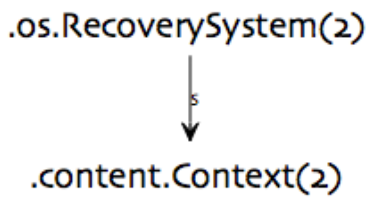
Ground Rule
Context, ContentResolver, ContentValues, OperationApplicationException

android.database:1(3)
ContentObserver, CursorWindow, ContentObservable
android.database:2(14)
sqlite.SQLiteOpenHelper, CursorJoiner, sqlite.SQLiteQuery, sqlite.SQLiteProgram, CursorJoiner$Result, sqlite.SQLiteStatement, Cursor, sqlite.SQLiteQueryBuilder, sqlite.SQLiteDatabase$CursorFactory, CrossProcessCursor, sqlite.SQLiteDatabase, DatabaseUtils, DatabaseUtils$InsertHelper, sqlite.SQLiteCursorDriver
android.database:3(3)
AbstractCursor$SelfContentObserver, AbstractCursor, CursorWrapper

Content - os



***android.os:2->android.content:2
RecoverySystem Context S



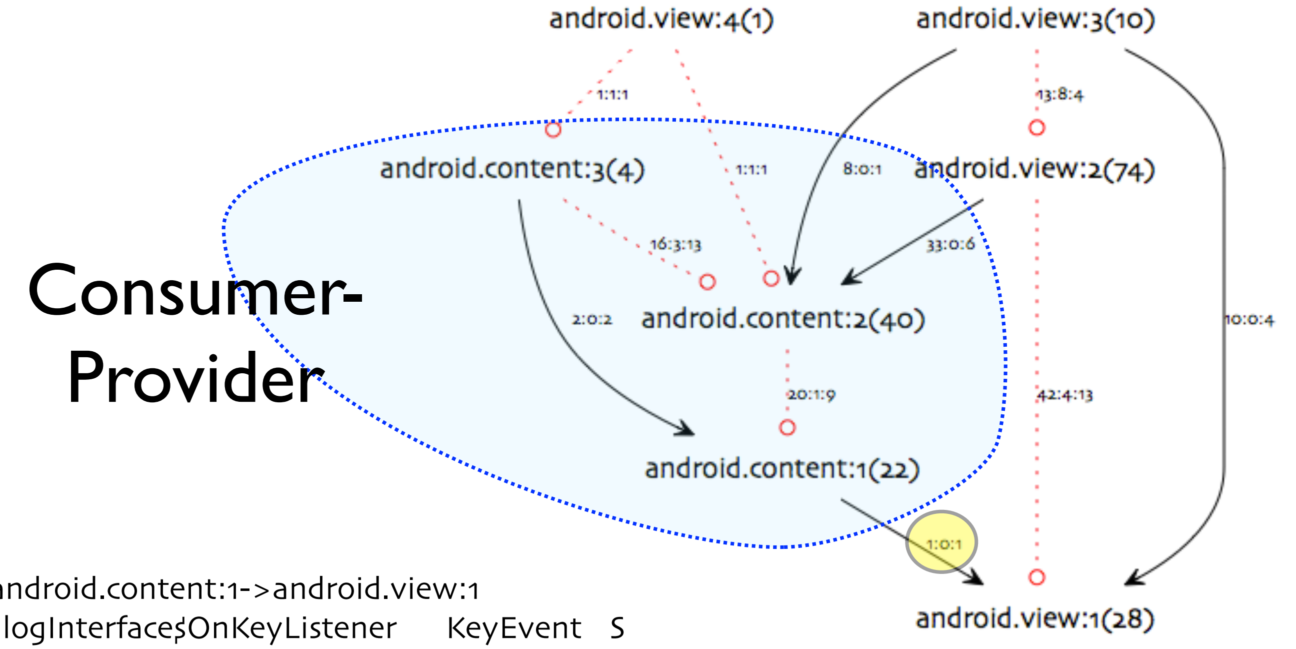
Ground Rule

RecoverySystem to Context?

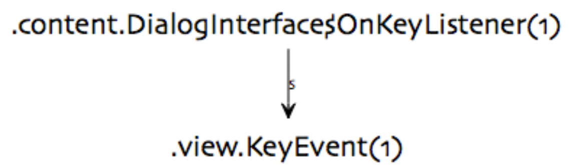
android.os:2(2)
RecoverySystem, RecoverySystem$ProgressListener

1 Edges

Content - view



***android.content:1->android.view:1
DialogInterface$OnKeyListener KeyEvent S

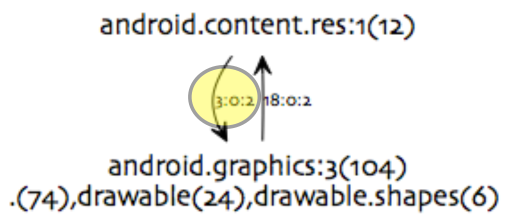


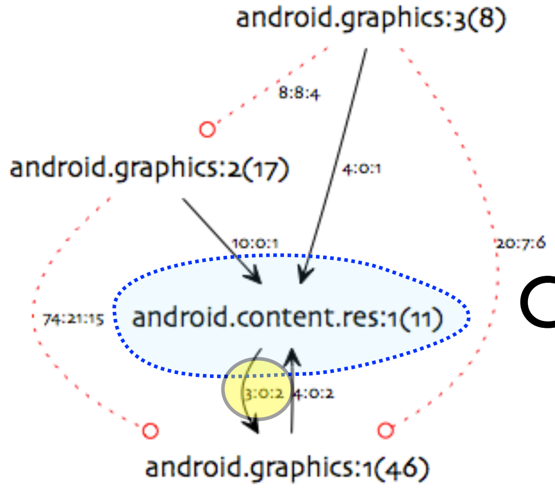
Violation

Why content should know key processing

1 Edges

Cycle 4. content.res

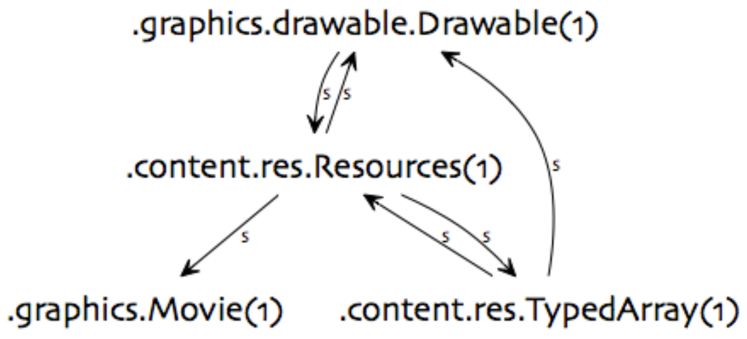

android.content.res:1(12)
3:0:2
18:0:2
android.graphics:3(104)
.(74),drawable(24),drawable.shapes(6)



Consumer-Provider

***android.content.res:1->android.graphics:1

Resources	drawable.Drawable	S
Resources	Movie	S
TypedArray	drawable.Drawable	S


.graphics.drawable.Drawable(1)
s
s
.content.res.Resources(1)
s
s
s
s
s
.graphics.Movie(1)
.content.res.TypedArray(1)

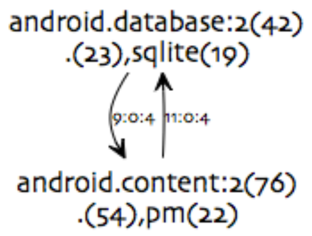
Violation

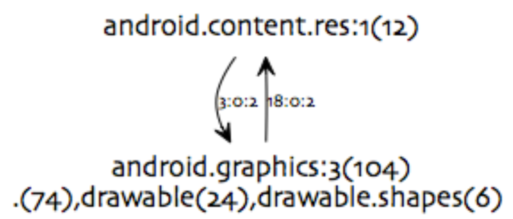
Why?

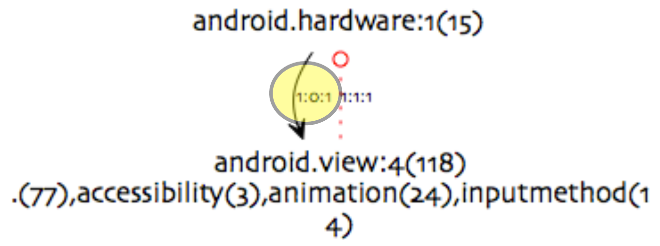
3 Edges

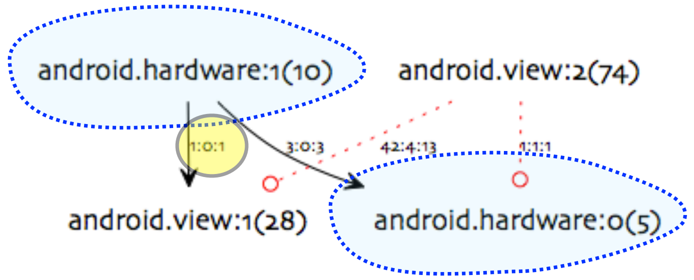
Cycle 5. hardware

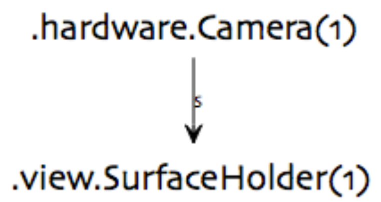
database, graphics are already dealt




android.content.res:1(12)
3:0:2
18:0:2
android.graphics:3(104)
.(74),drawable(24),drawable.shapes(6)







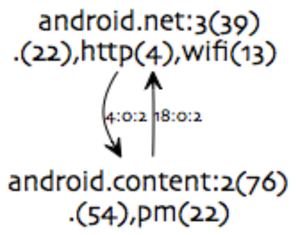
Violation

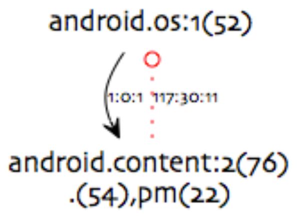
Why?

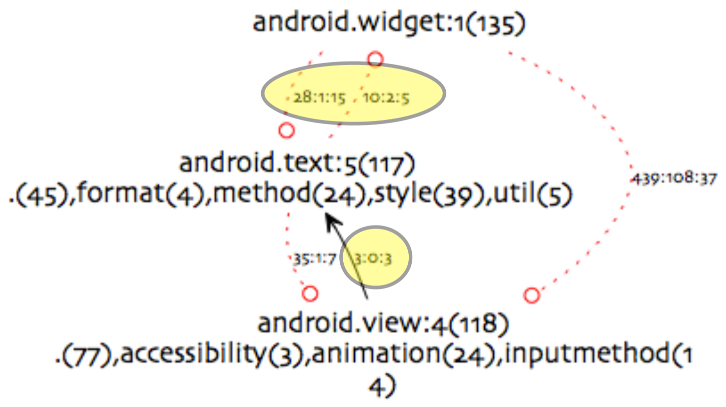
1 Edges

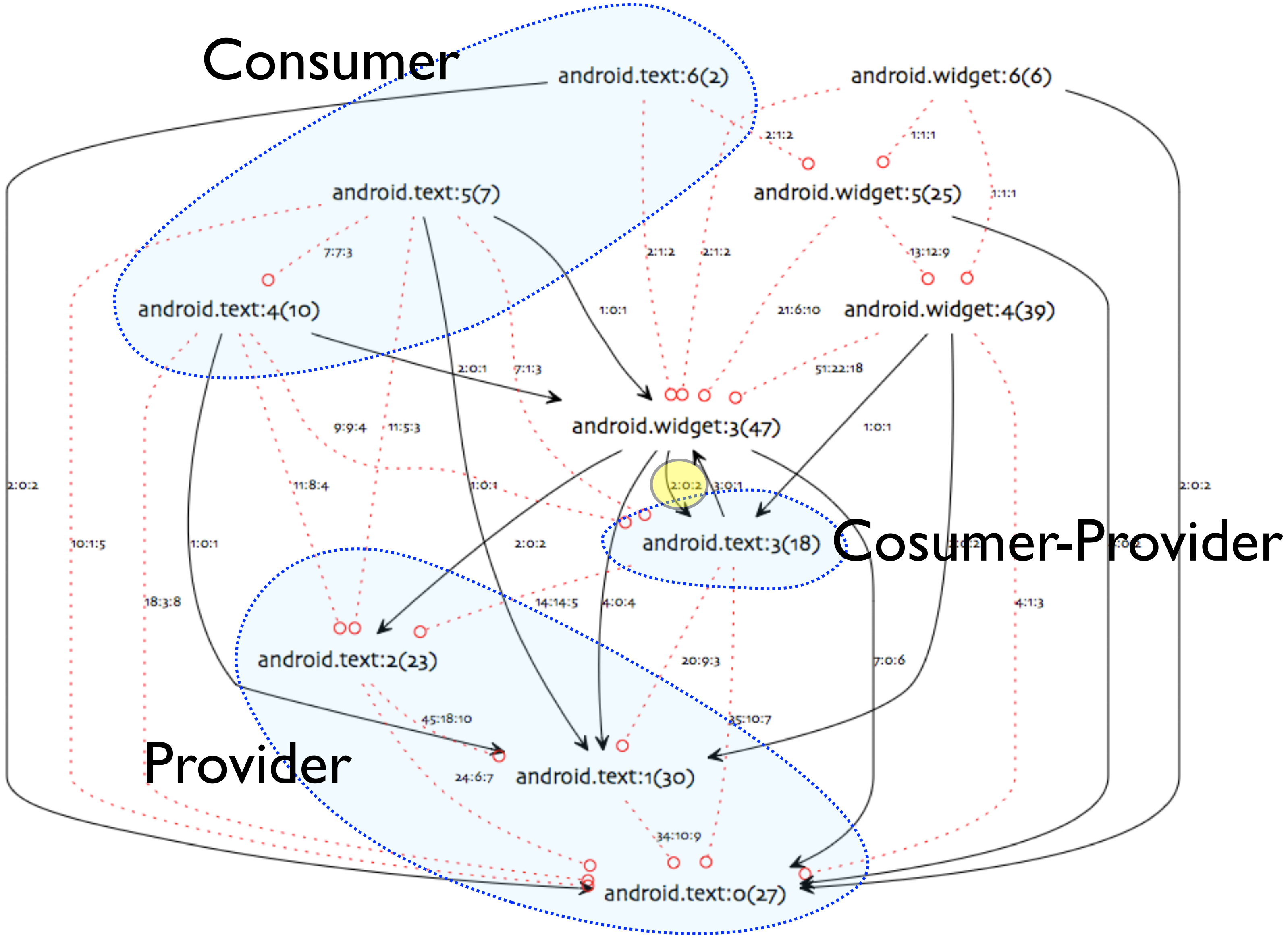
Cycle 6. text

net, os are already dealt








Consumer
android.text:6(2)
android.widget:6(6)
android.text:5(7)
android.widget:5(25)
android.text:4(10)
android.widget:4(39)
android.widget:3(47)
android.text:3(18)
Cosumer-Provider
android.text:2(23)
android.text:1(30)
Provider
android.text:0(27)
2:1:2
1:1:1
7:7:3
2:1:2
2:1:2
13:12:9
1:0:1
21:6:10
2:0:1
7:1:3
51:22:18
9:9:4
11:5:3
1:0:1
11:8:4
1:0:1
2:0:2
3:0:1
2:0:2
2:0:2
10:1:5
1:0:1
2:0:2
18:3:8
14:14:5
4:0:4
4:1:3
20:9:3
7:0:6
45:18:10
25:10:7
24:6:7
34:10:9

MVC

- text model & controller in text package
- widget as a view
- divide text into model & controller part
- dependency: C -> V -> M 로

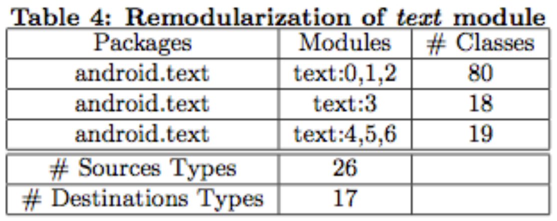
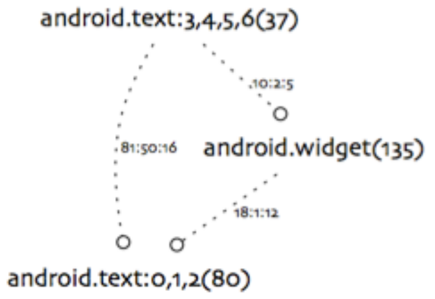


Table 4: Remodularization of *text* module

Packages	Modules	# Classes
android.text	text:0,1,2	80
android.text	text:3	18
android.text	text:4,5,6	19
# Sources Types	26	
# Destinations Types	17	



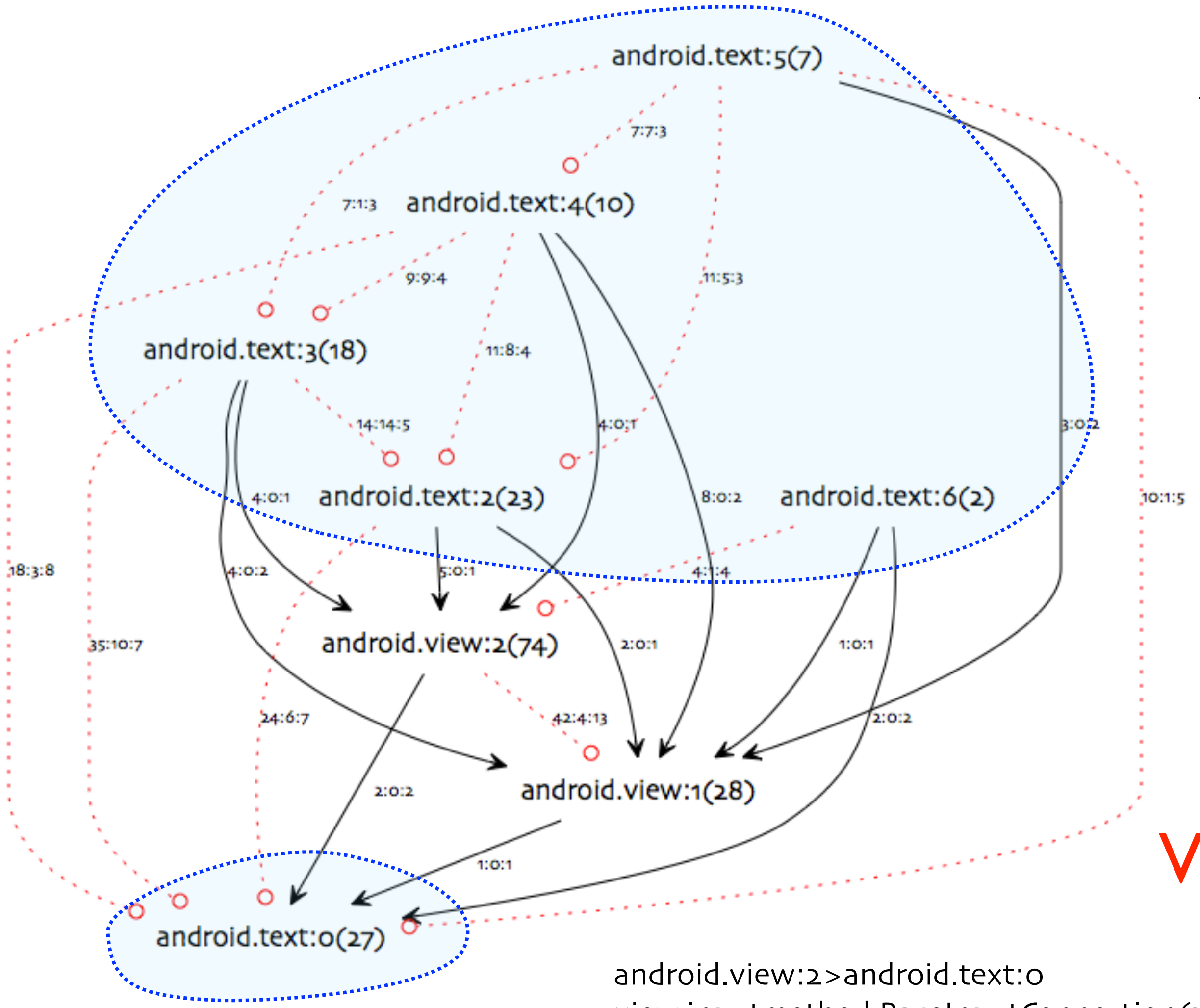
Ignore problematic edge
(widget:3,text:3)
android.widget:3>android.text:3
widget.TextView(3) -> text.method.MovementMethod(3): S
widget.TextView(3) -> text.style.URLSpan(3): S

text3 as Client so that text:3,4,5,6

(text:4, 5, 6 -> widget:3,5)
TextView
AdapterView
MultiAutoCompleteTextView

(widget:3, 4, 5, 6 -> text:0, 1, 3)
TextUtils, Layout, TextPaint
Editable, TextWatcher, NoCopySpan
Spannable, Editable, InputFilter
method.MovementMethod, style.URLSpan
method.KeyListener, method.TransformationMethod

3 Edges



Violation? Rule?

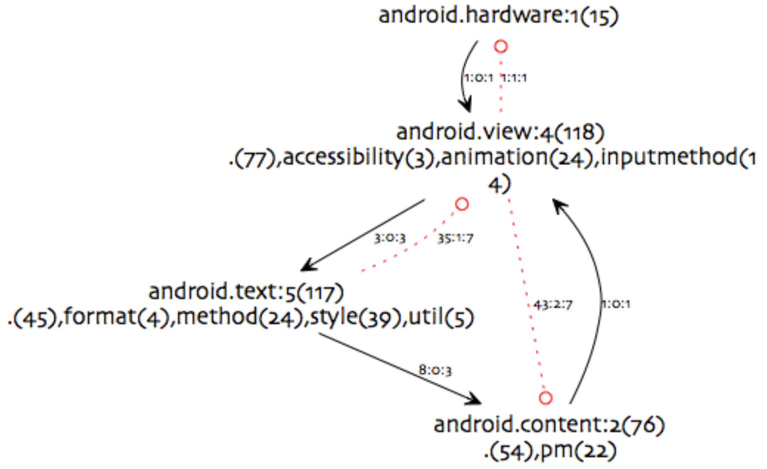
android.view:2>android.text:0
view.inputmethod.BaseInputConnection(2) -> text.Spannable(0): S
view.inputmethod.BaseInputConnection(2) -> text.Editable(0): S

(view:1, 2 -> text:0)
InputType, Spannable, Editable

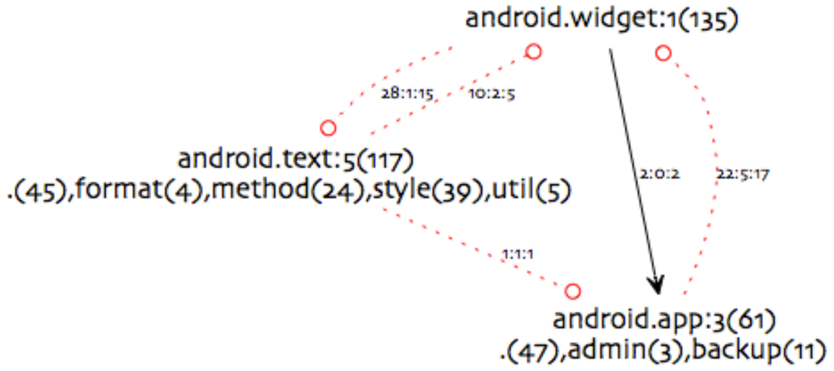
android.view:1>android.text:0
view.inputmethod.EditorInfo(1) -> text.InputType(0): S,P

3 Edges

view and widget are already dealt



android.hardware:1(15)
1:0:1
1:1:1
android.view:4(118)
.(77),accessibility(3),animation(24),inputmethod(14)
3:0:3
35:1:7
43:2:7
1:0:1
android.text:5(117)
.(45),format(4),method(24),style(39),util(5)
8:0:3
android.content:2(76)
.(54),pm(22)



android.widget:1(135)
28:1:15
10:2:5
2:0:2
22:5:17
android.text:5(117)
.(45),format(4),method(24),style(39),util(5)
1:1:1
android.app:3(61)
.(47),admin(3),backup(11)