Automating GUI Testing for Android Applications

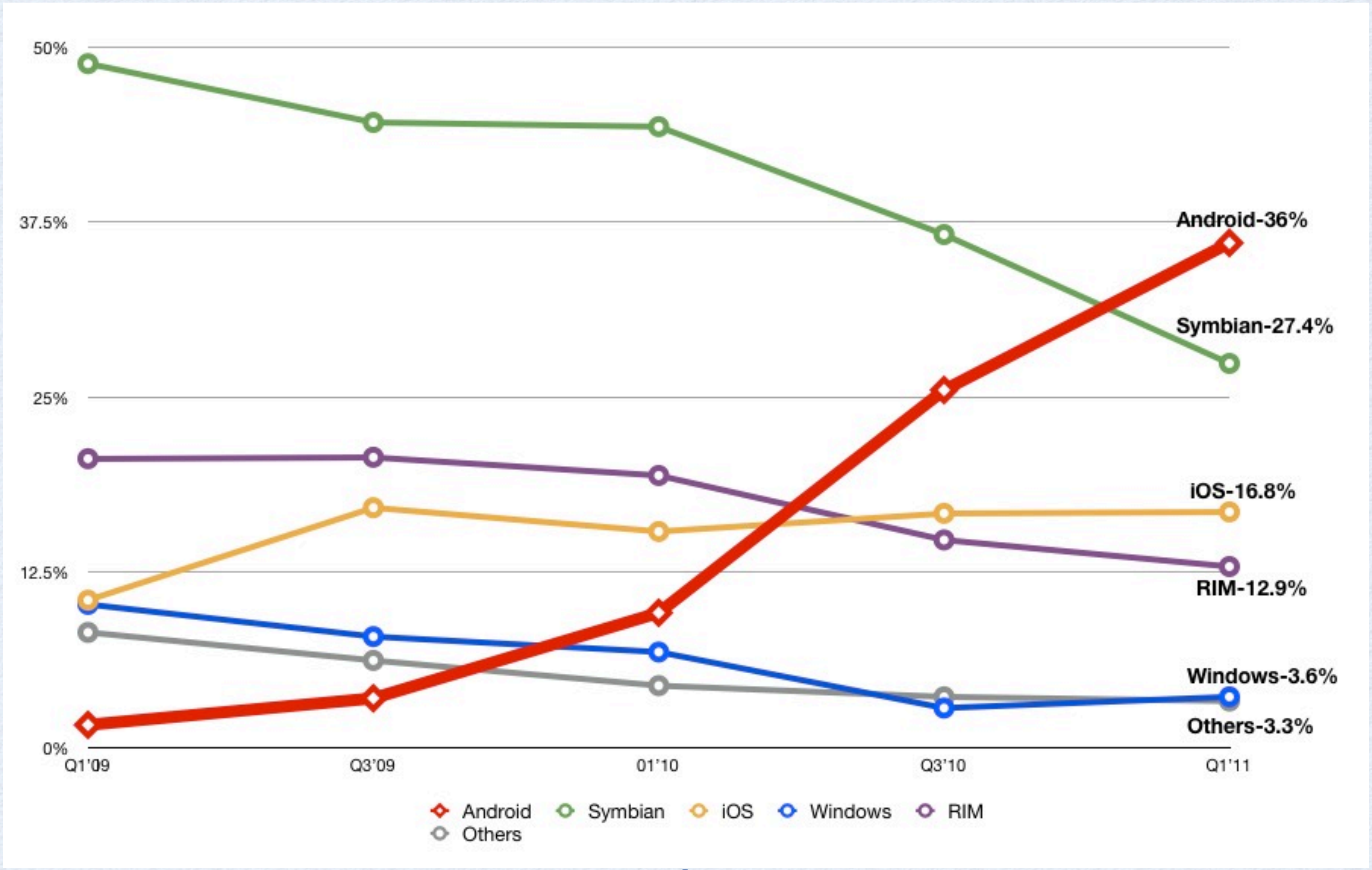
Cuixiong(Tony) Hu

Amazon Inc.

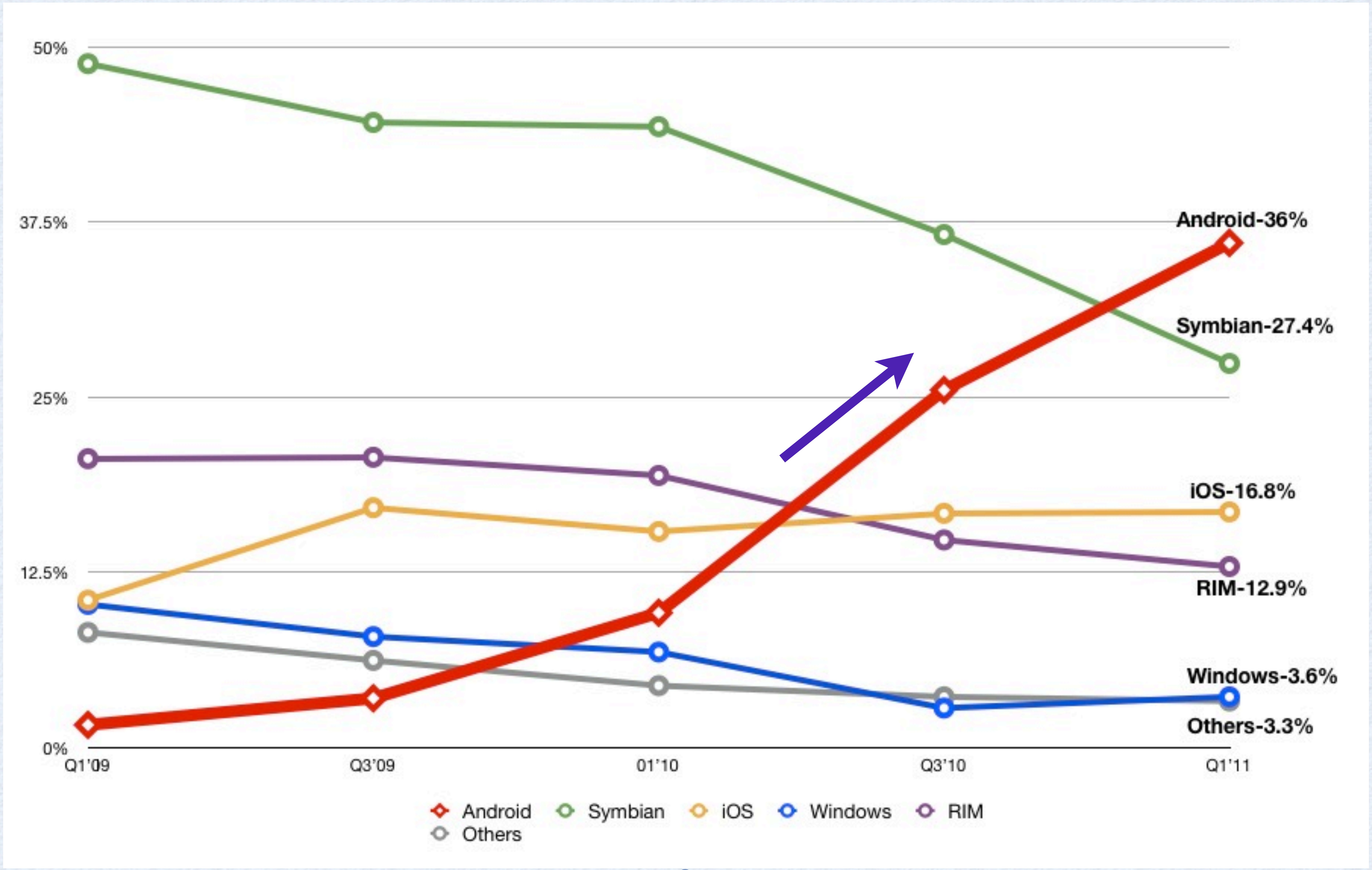
Iulian Neamtiu

University of California, Riverside

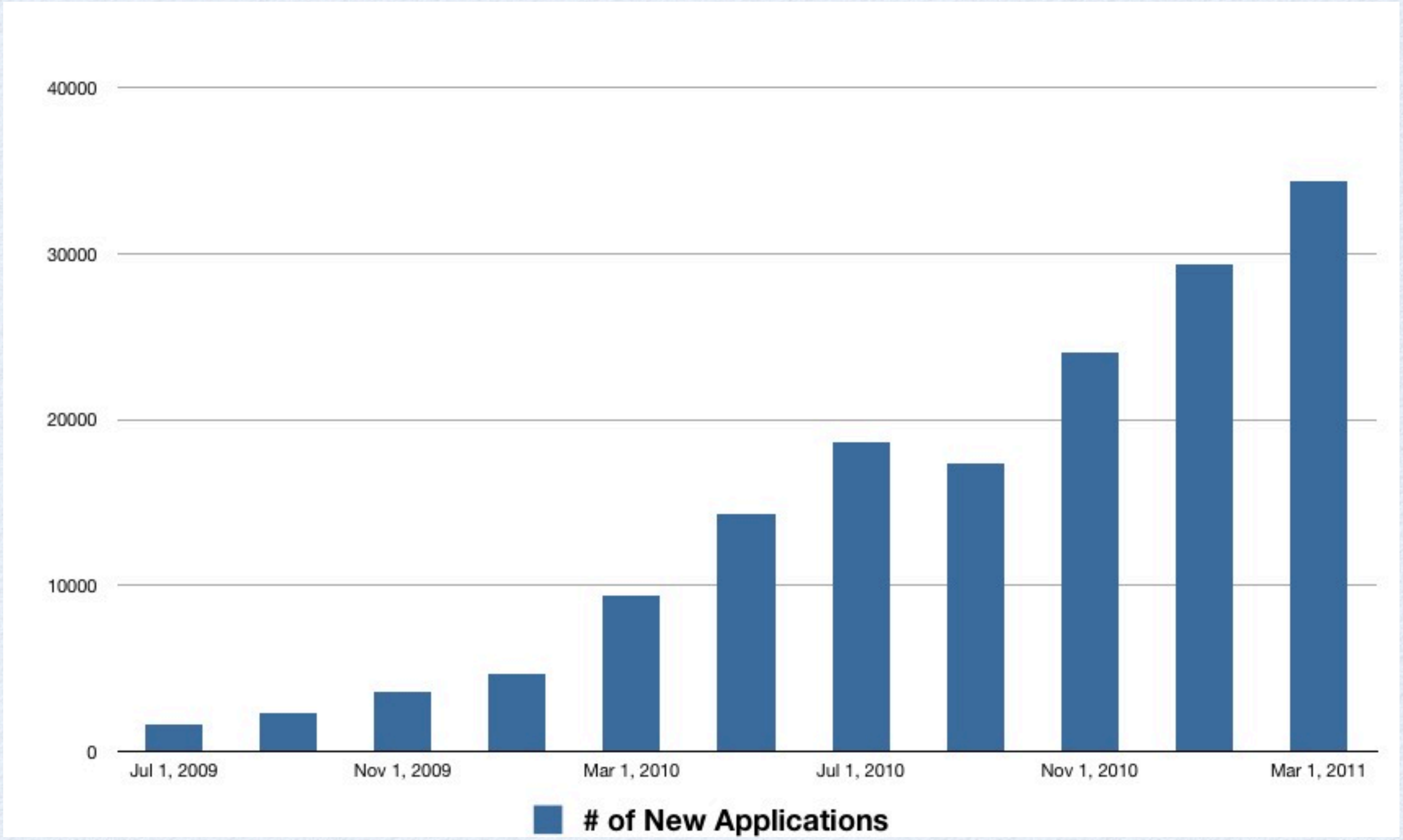
Android Gaining Market Share



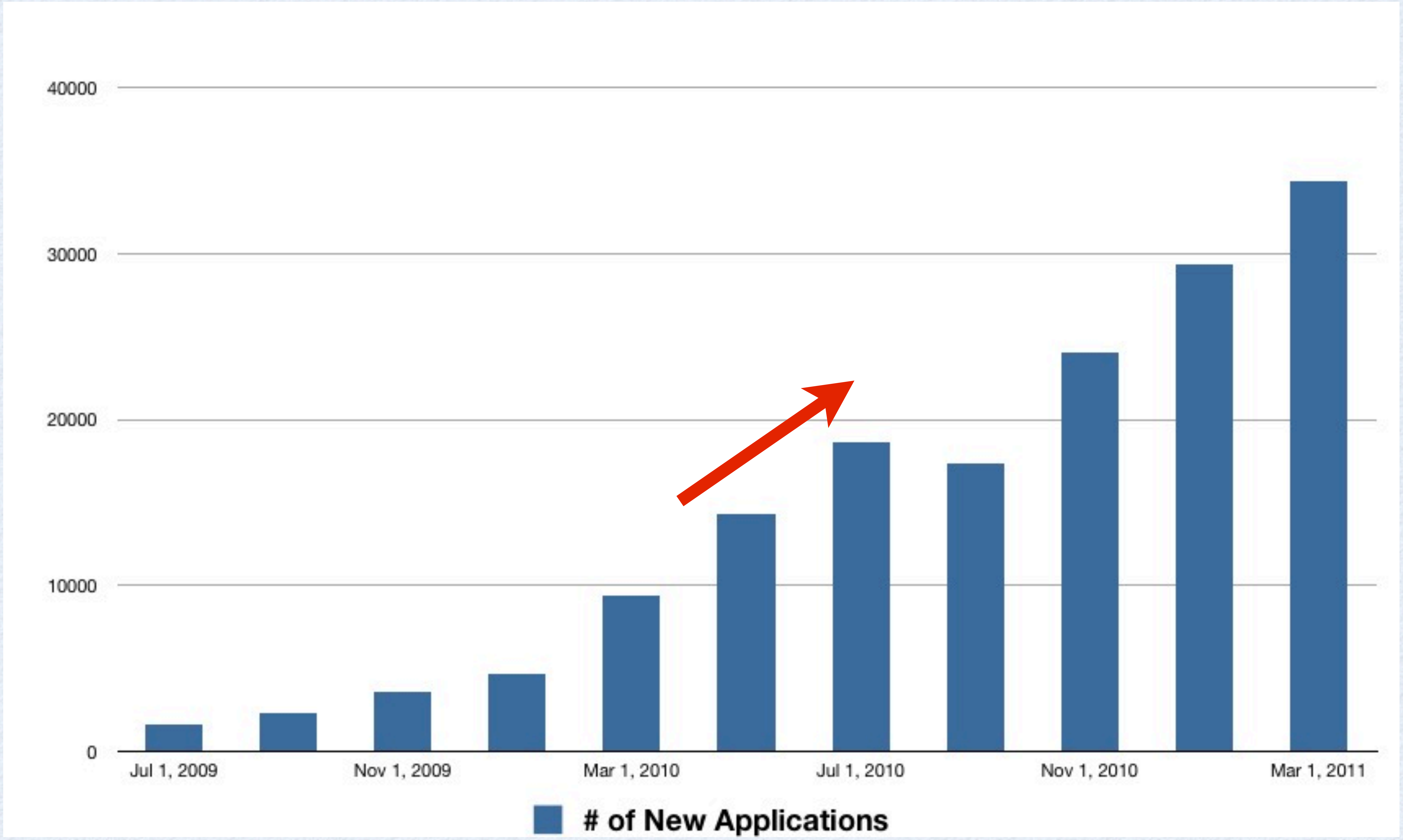
Android Gaining Market Share



More and More New Android Applications Released



More and More New Android Applications Released



Motivation

Motivation

- ✦Android platform gaining traction

Motivation

✦Android platform gaining traction

→ Correctness of Android applications is CRITICAL (mobile banking, business mgmt.)

Motivation

✦Android platform gaining traction

➡ Correctness of Android applications is CRITICAL (mobile banking, business mgmt.)

What's novel about the Android platform?

Motivation

✦Android platform gaining traction

⇨ Correctness of Android applications is CRITICAL (mobile banking, business mgmt.)

✦Novelty of Android development model

What's novel about the Android platform?

Motivation

✦Android platform gaining traction

⇨ Correctness of Android applications is CRITICAL (mobile banking, business mgmt.)

✦Novelty of Android development model

⇨ **Android-specific bug study**

What's novel about the Android platform?

Motivation

- ✦Android platform gaining traction
  - Correctness of Android applications is CRITICAL (mobile banking, business mgmt.)
- ✦Novelty of Android development model
  - **Android-specific bug study**
- ✦Android-specific verification tools

What's novel about the Android platform?

Motivation

✦Android platform gaining traction

→ Correctness of Android applications is CRITICAL (mobile banking, business mgmt.)

✦Novelty of Android development model

→ **Android-specific bug study**

✦Android-specific verification tools

→ **Automated approach for test generation and runtime verification for Android applications**

What's novel about the Android platform?

What is Android

- Open source platform for mobile devices, started October 2008
- A complete software stack
  - OS
  - Middleware
  - Applications
- Based on Java

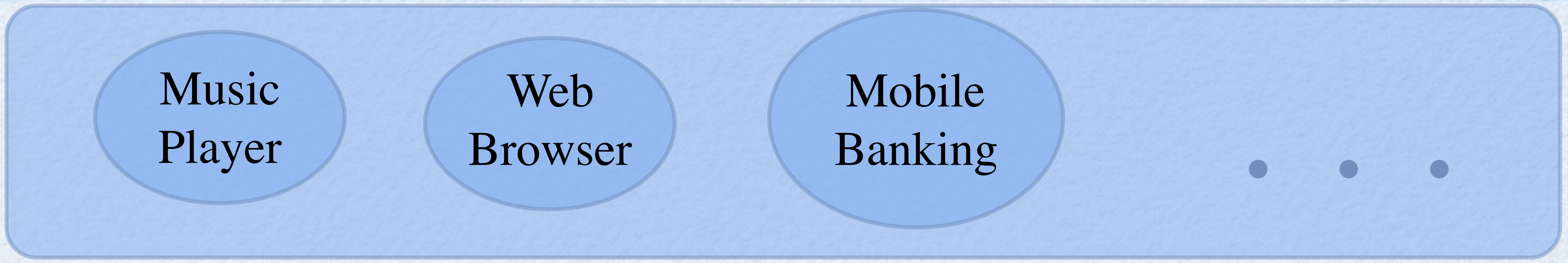
Components of Android Applications

- Activity
    - Control application windows
- Services
    - Run in the background for an indefinite period of time
- Broadcast Receivers
    - React to a broadcast information, i.e., low battery
- Content Providers
    - Store content, allow sharing with other apps

Architecture of Android

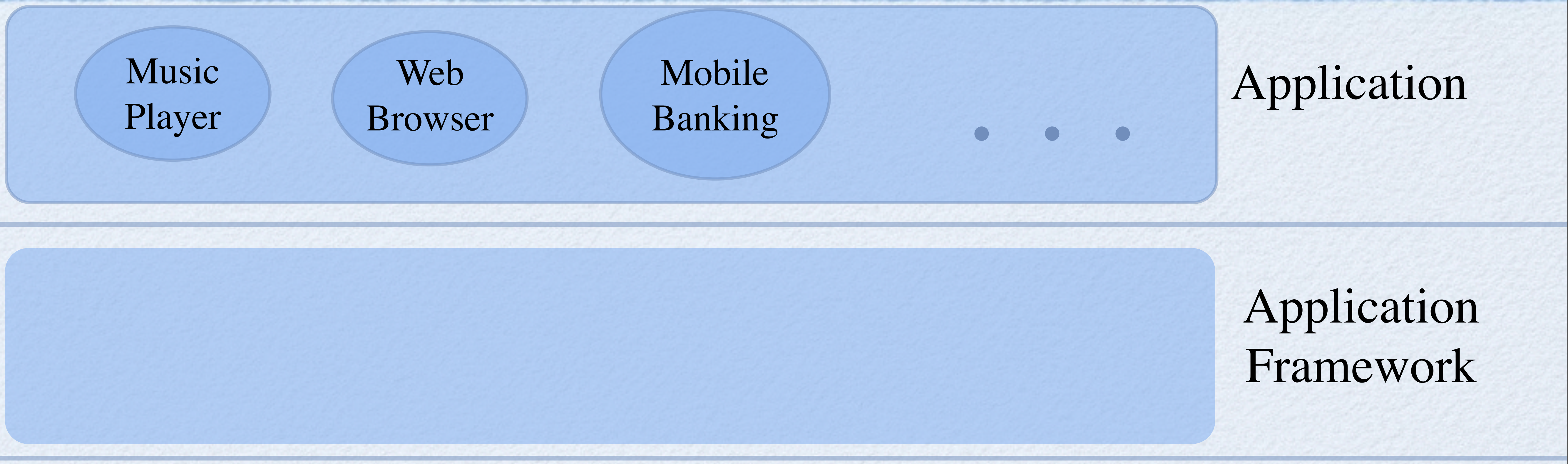
Architecture of Android

Architecture of Android

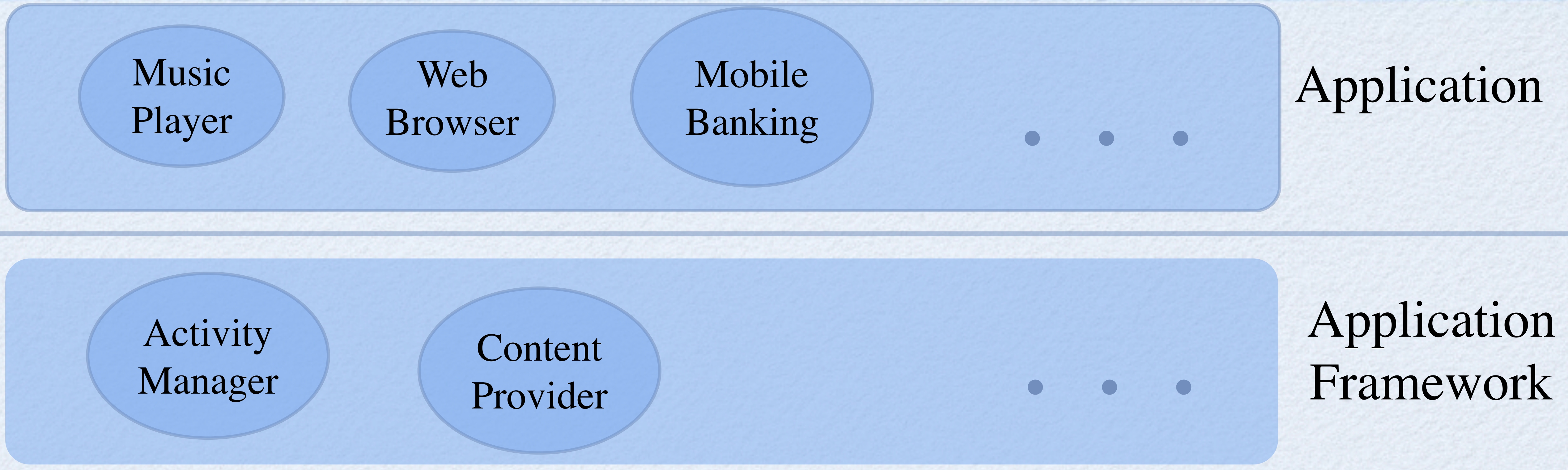


Application

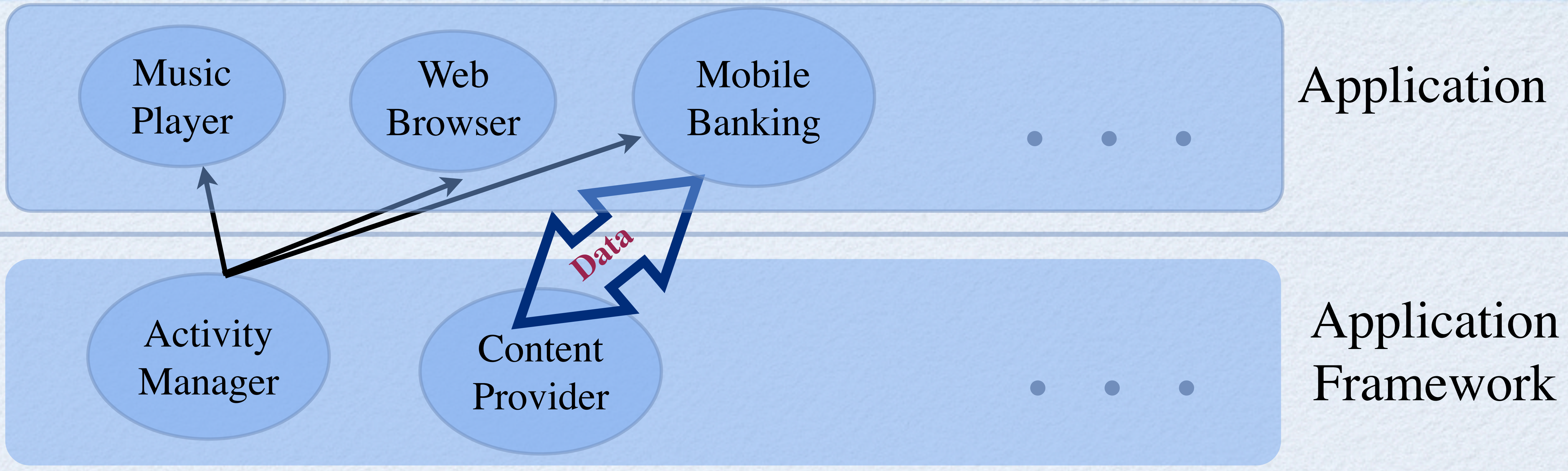
Architecture of Android



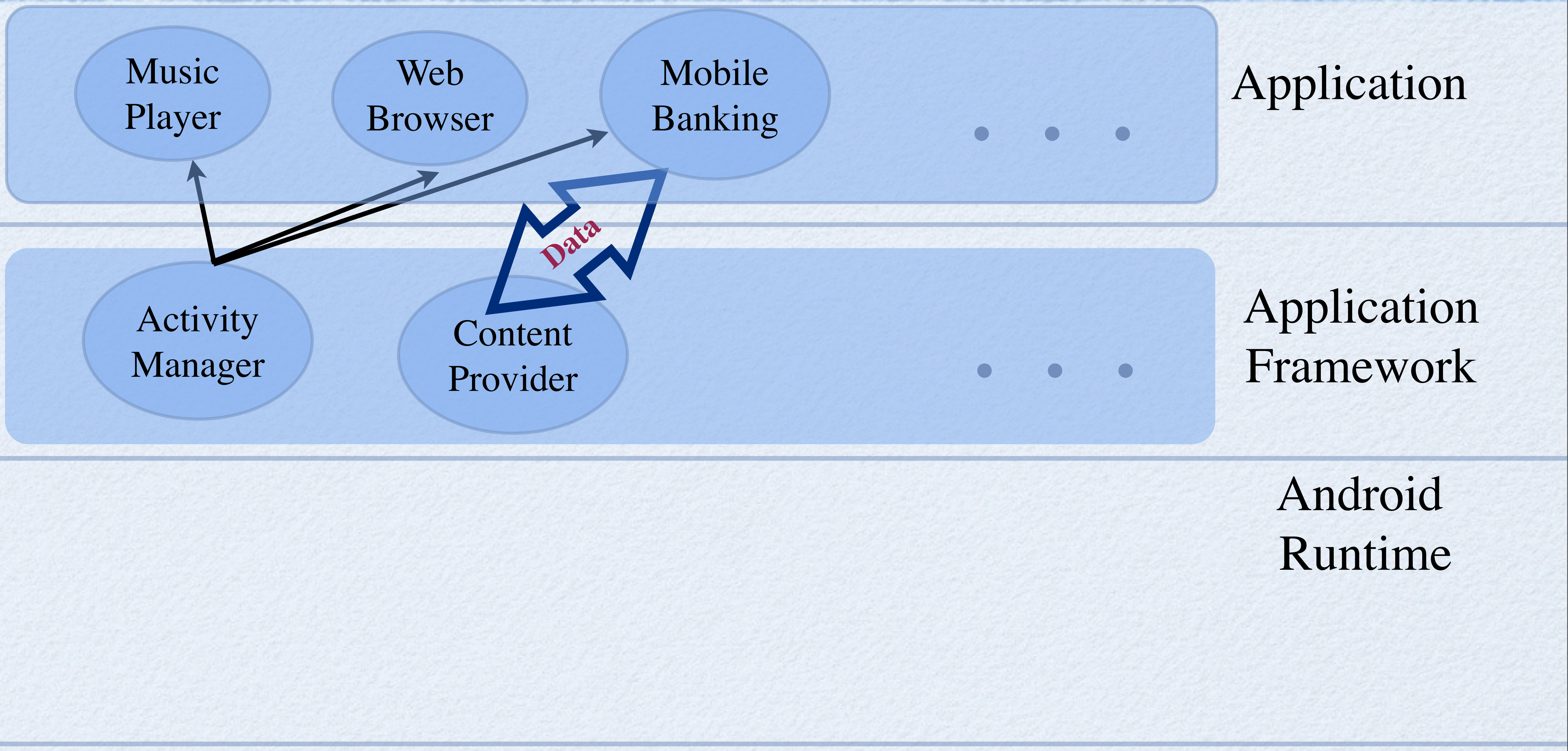
Architecture of Android



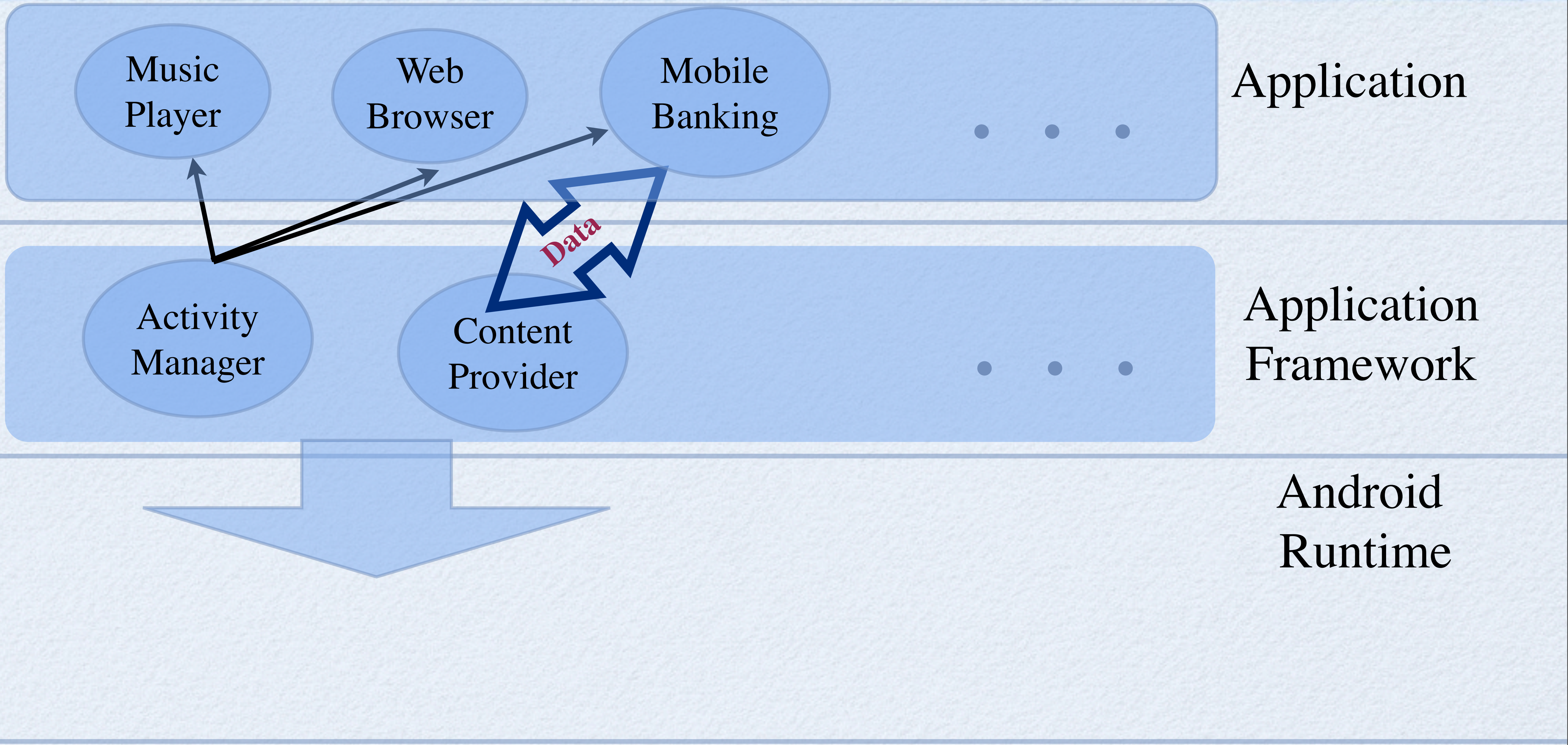
Architecture of Android



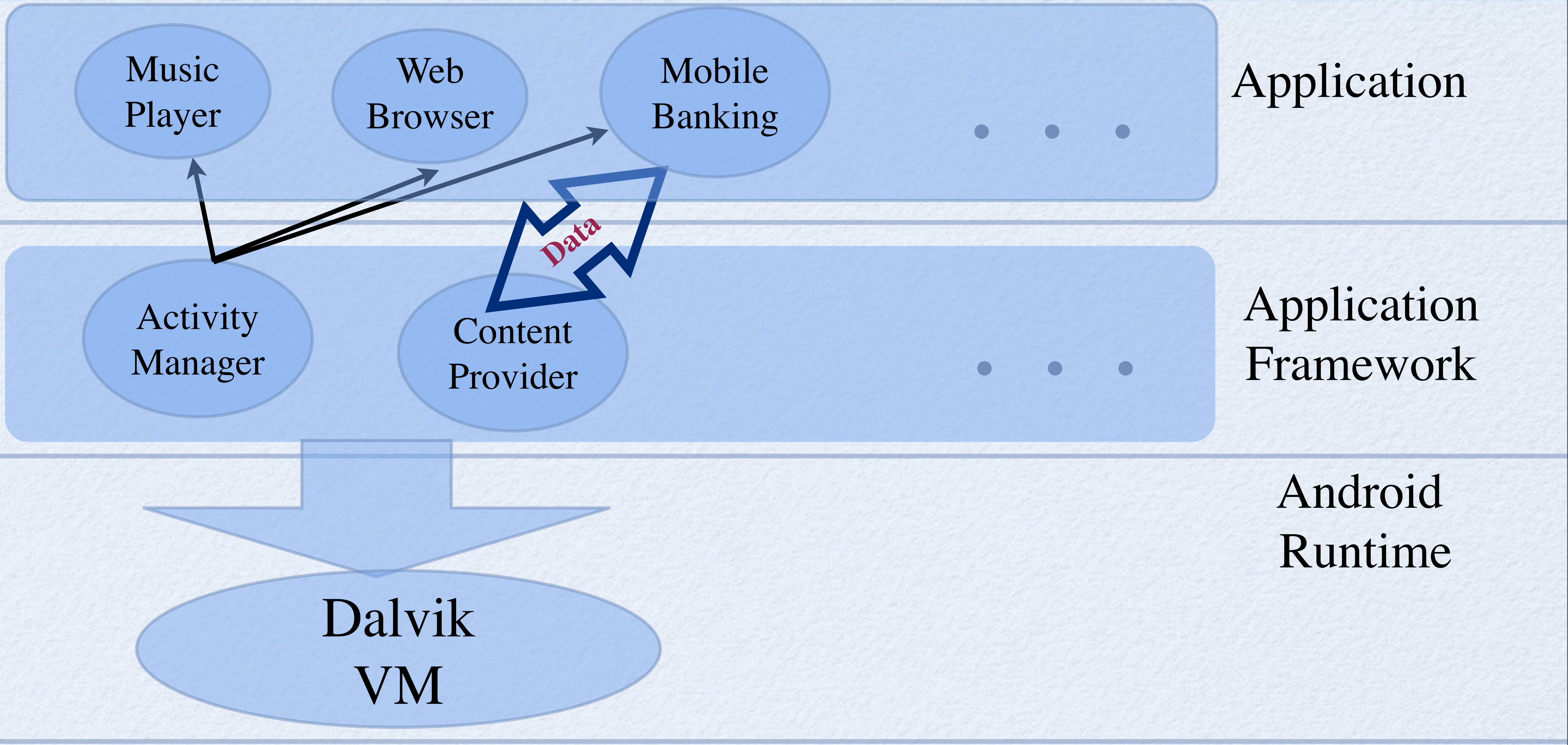
Architecture of Android



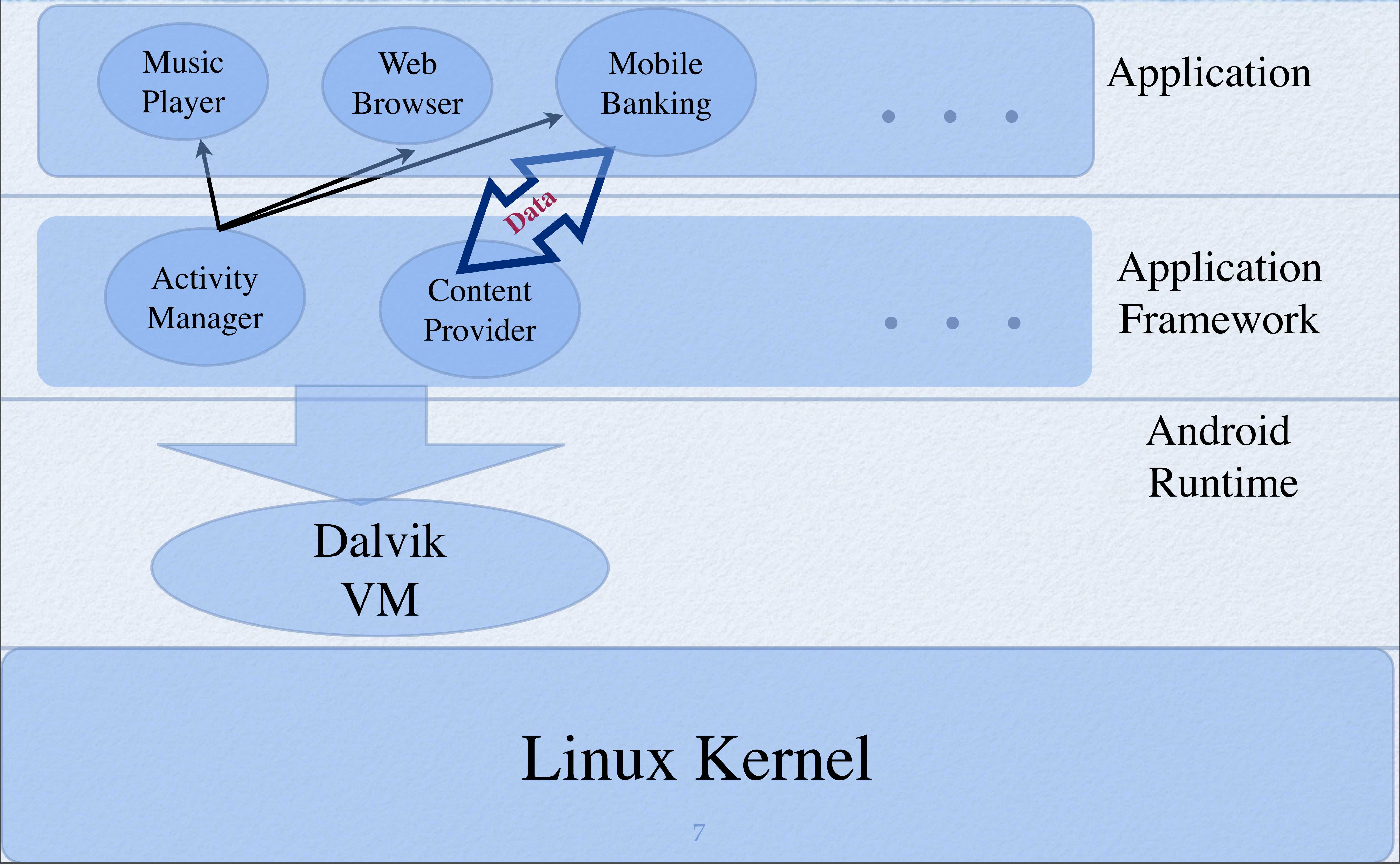
Architecture of Android



Architecture of Android



Architecture of Android



Roadmap

Roadmap

Android development model is unique

Roadmap

Android development model is unique

$\Downarrow$

Android-specific, novel kinds of bugs

Roadmap

Android development model is unique

Android-specific, novel kinds of bugs

Bug study and bug categorization

Applications Examined

Program	First Release	# of updates	# of bugs
Andoku	06/2009	382	1
GuessTheNumber	02/2009	71	2
Delicious	02/2009	60	5
MonolithAndroid	11/2008	167	7
Opensudoku	04/2009	393	7
CMIS	01/2010	41	8
DealDroid	03/2009	164	10
Skylight1	09/2009	709	10
Rokon	09/2009	362	29
ConnectBot	08/2008	508	79

Bug Study Results

Program	Bug Category								
	Activity	*Event*	*Type*	*Unhandled Exception*	*API*	*I/O*	*Concurr-ency*	*Other*	**Total**
Skylight1	3	2	0	1	0	0	0	4	**10**
CMIS	0	0	0	2	0	0	0	6	**8**
Delicious	0	0	0	0	1	0	0	4	**5**
ConnectBot	2	8	2	5	1	3	1	57	**79**
DealDroid	1	1	0	0	0	0	0	8	**10**
Rokon	0	6	2	3	0	4	0	14	**29**
Andoku	0	0	0	0	0	0	0	1	**1**
Opensudoku	1	1	0	0	0	0	0	5	**7**
GuessTheNumber	1	1	0	0	0	0	0	0	**2**
MonolithAndroid	0	2	0	0	2	0	0	3	**7**
Total	**8**	**21**	**4**	**11**	**4**	**7**	**1**	**102**	**158**

Bug Study Results

Program	Bug Category								
	Activity	*Event*	*Type*	*Unhandled Exception*	*API*	*I/O*	*Concurr-ency*	*Other*	**Total**
Skylight1	3	2	0	1	0	0	0	4	**10**
CMIS	0	0	0	2	0	0	0	6	**8**
Delicious	0	0	0	0	1	0	0	4	**5**
ConnectBot	2	8	2	5	1	3	1	57	**79**
DealDroid	1	1	0	0	0	0	0	8	**10**
Rokon	0	6	2	3	0	4	0	14	**29**
Andoku	0	0	0	0	0	0	0	1	**1**
Opensudoku	1	1	0	0	0	0	0	5	**7**
GuessTheNumber	1	1	0	0	0	0	0	0	**2**
MonolithAndroid	0	2	0	0	2	0	0	3	**7**
Total	**8**	**21**	**4**	**11**	**4**	**7**	**1**	**102**	**158**

Bug Study Results

Program	How to detect?			Bug Category					
	Activity	*Event*	*Type*	*Unhandled Exception*	*API*	*I/O*	*Concurr-ency*	*Other*	**Total**
Skylight1	3	2	0	1	0	0	0	4	**10**
CMIS	0	0	0	2	0	0	0	6	**8**
Delicious	0	0	0	0	1	0	0	4	**5**
ConnectBot	2	8	2	5	1	3	1	57	**79**
DealDroid	1	1	0	0	0	0	0	8	**10**
Rokon	0	6	2	3	0	4	0	14	**29**
Andoku	0	0	0	0	0	0	0	1	**1**
Opensudoku	1	1	0	0	0	0	0	5	**7**
GuessTheNumber	1	1	0	0	0	0	0	0	**2**
MonolithAndroid	0	2	0	0	2	0	0	3	**7**
Total	**8**	**21**	**4**	**11**	**4**	**7**	**1**	**102**	**158**

Testing Techniques

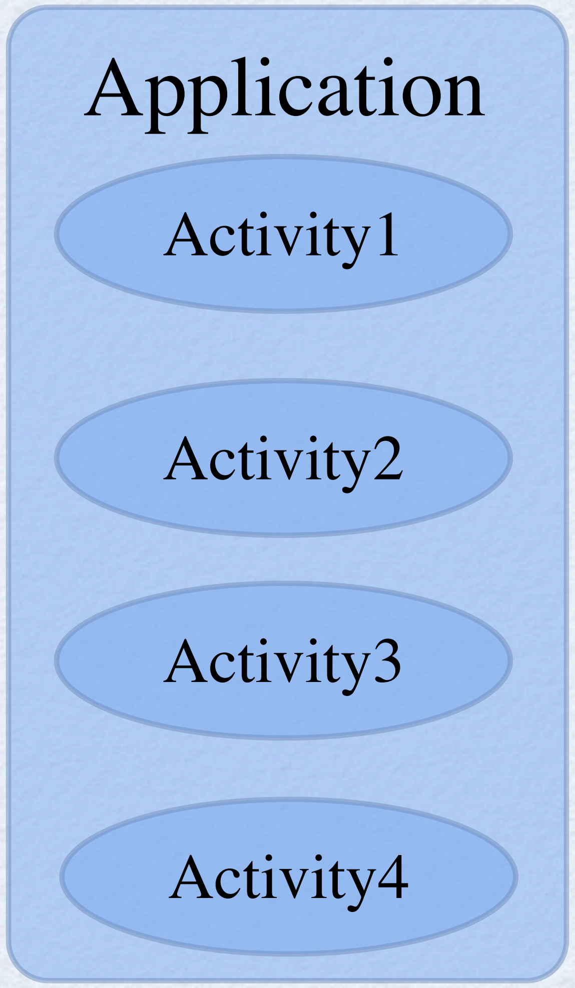
Combination of multiple methods:

1. Automatic event generation
2. Test case generation
3. Post-run application performance monitoring

Testing Techniques

Combination of multiple methods:

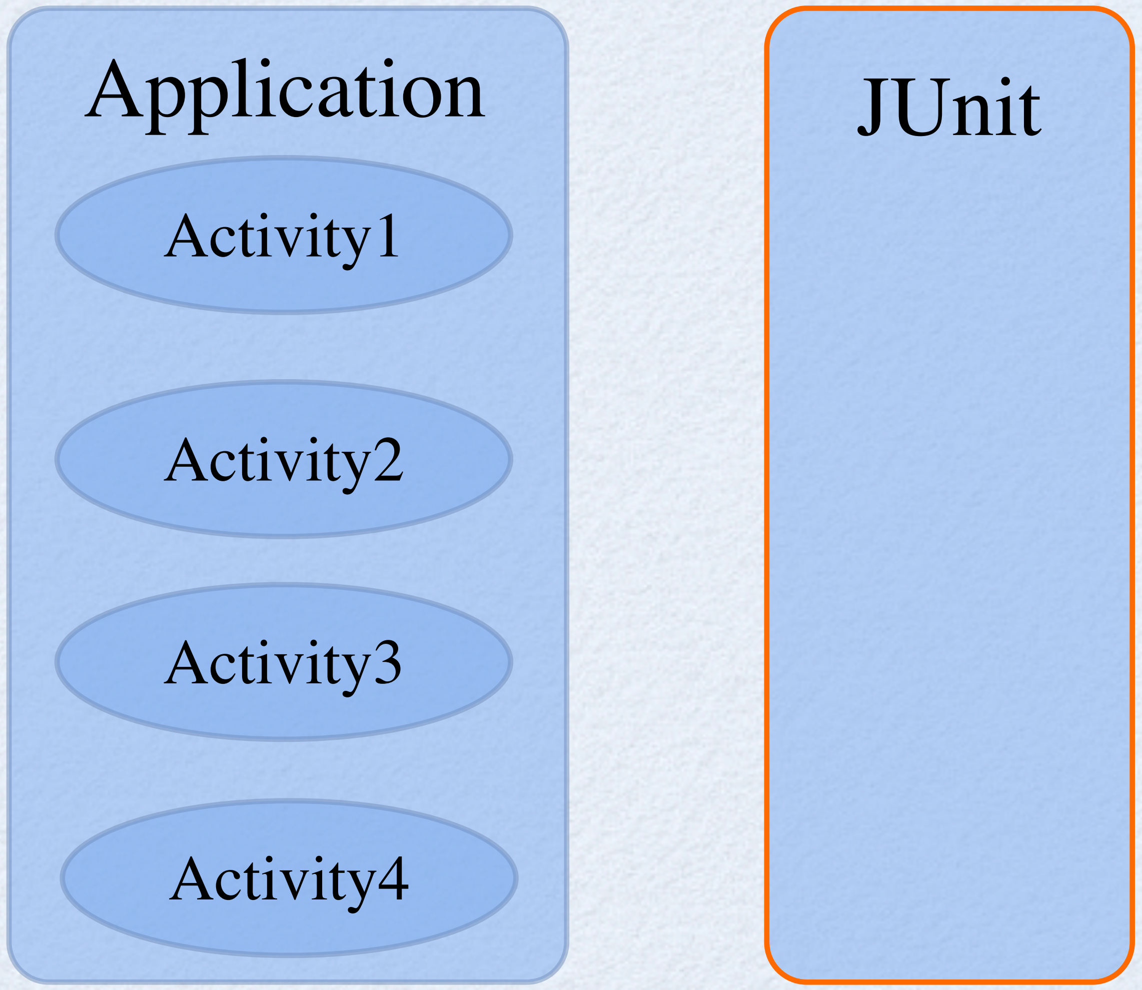
1. Automatic event generation
2. Test case generation
3. Post-run application performance monitoring



Testing Techniques

Combination of multiple methods:

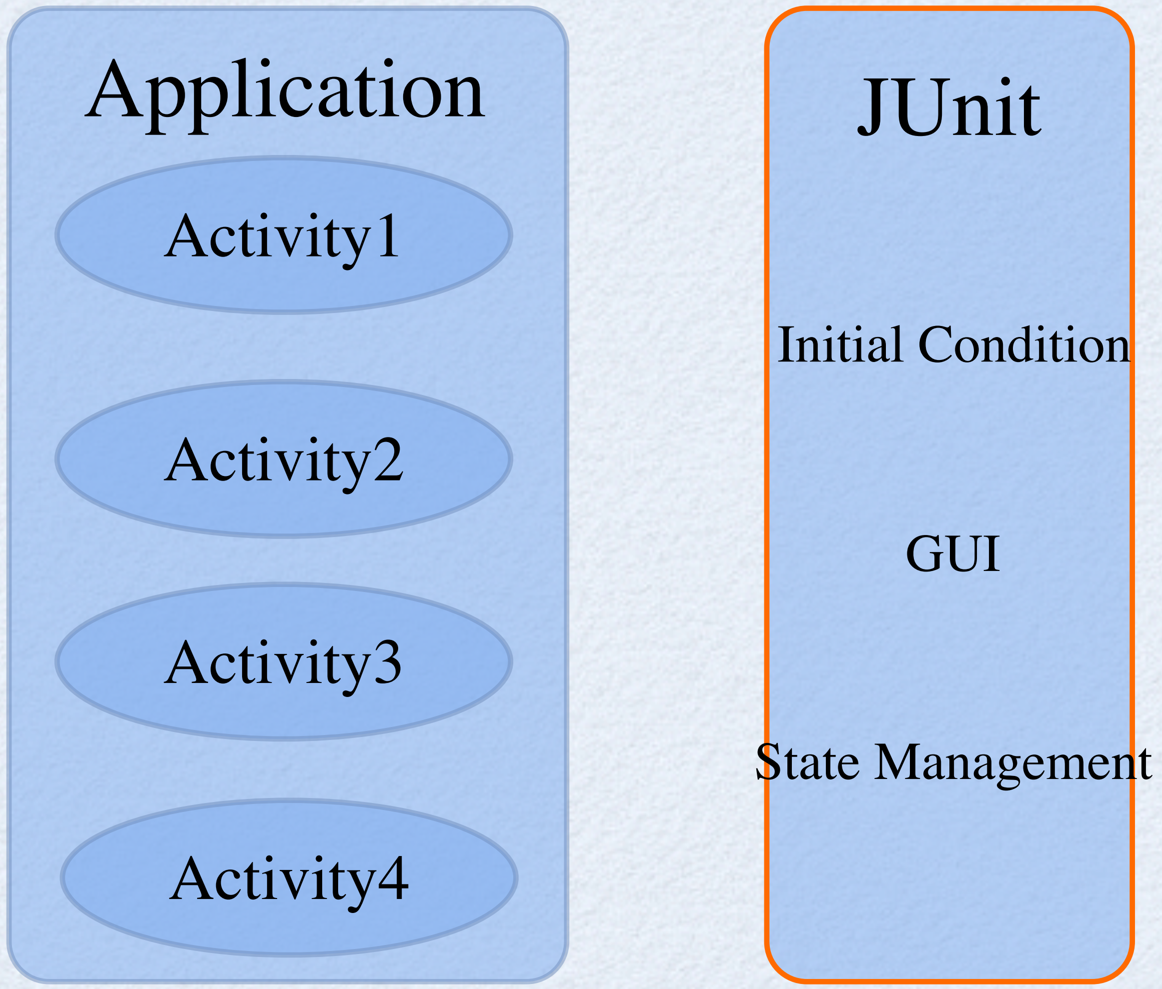
1. Automatic event generation
2. Test case generation
3. Post-run application performance monitoring



Testing Techniques

Combination of multiple methods:

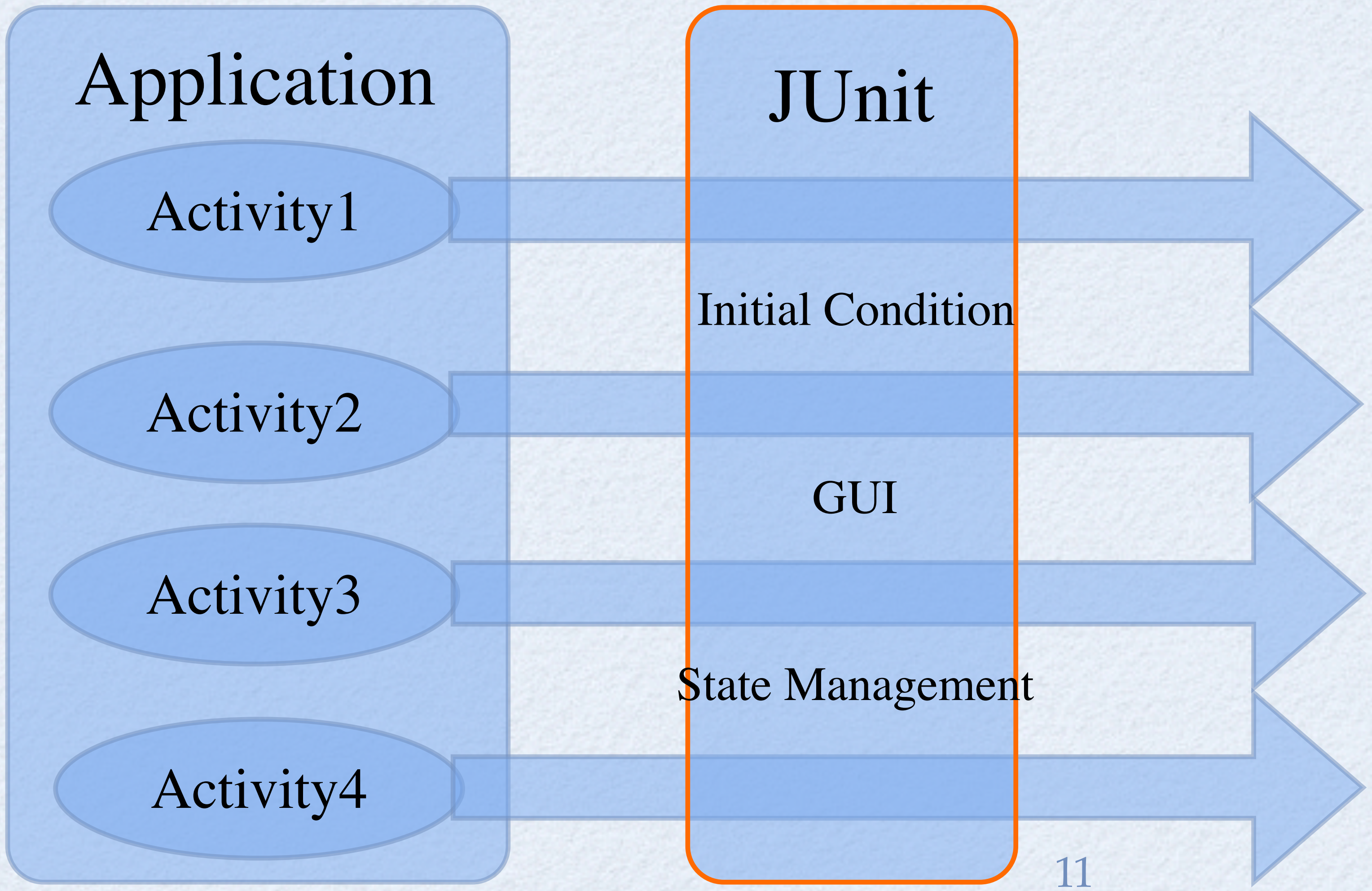
1. Automatic event generation
2. Test case generation
3. Post-run application performance monitoring



Testing Techniques

Combination of multiple methods:

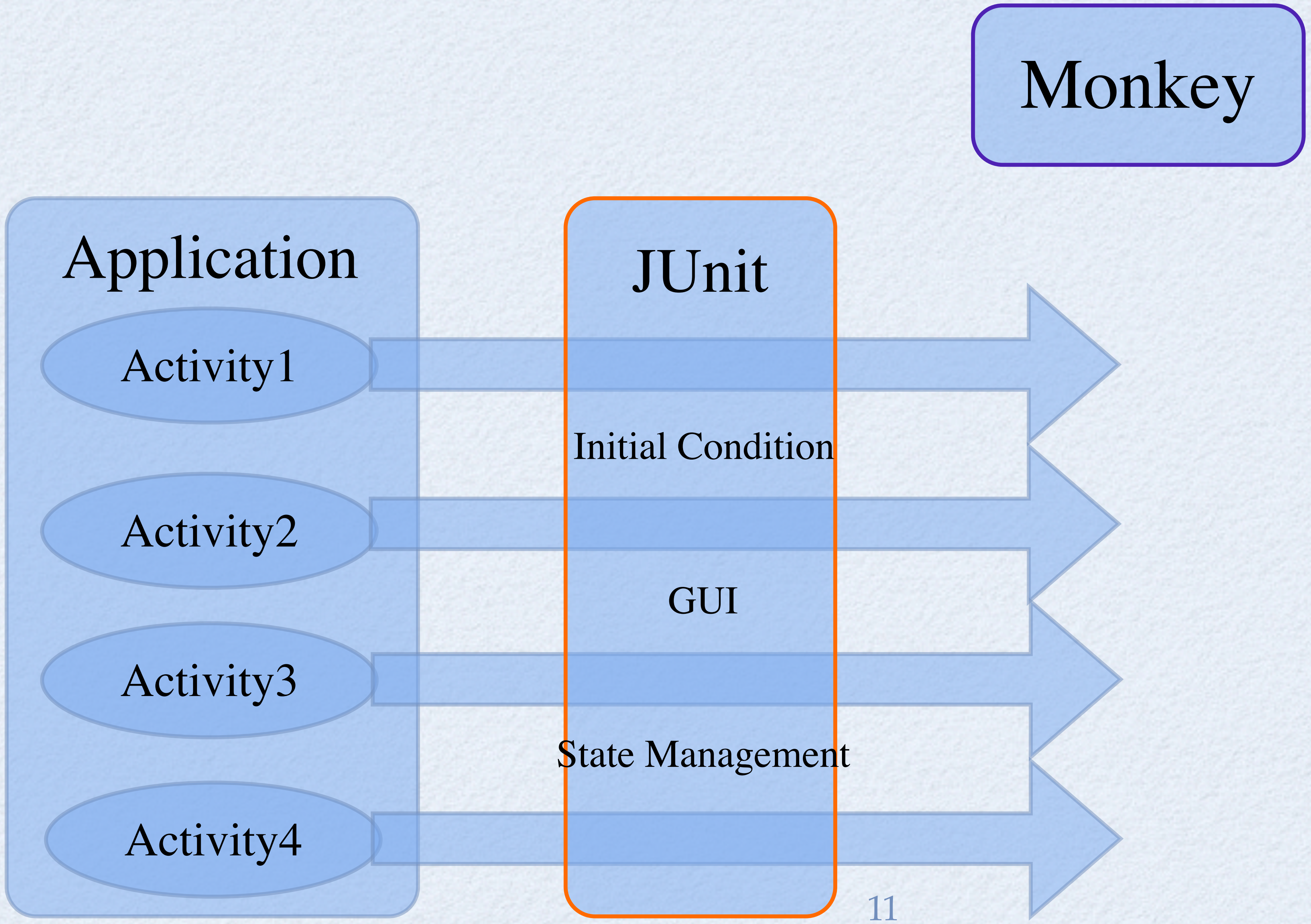
1. Automatic event generation
2. Test case generation
3. Post-run application performance monitoring



Testing Techniques

Combination of multiple methods:

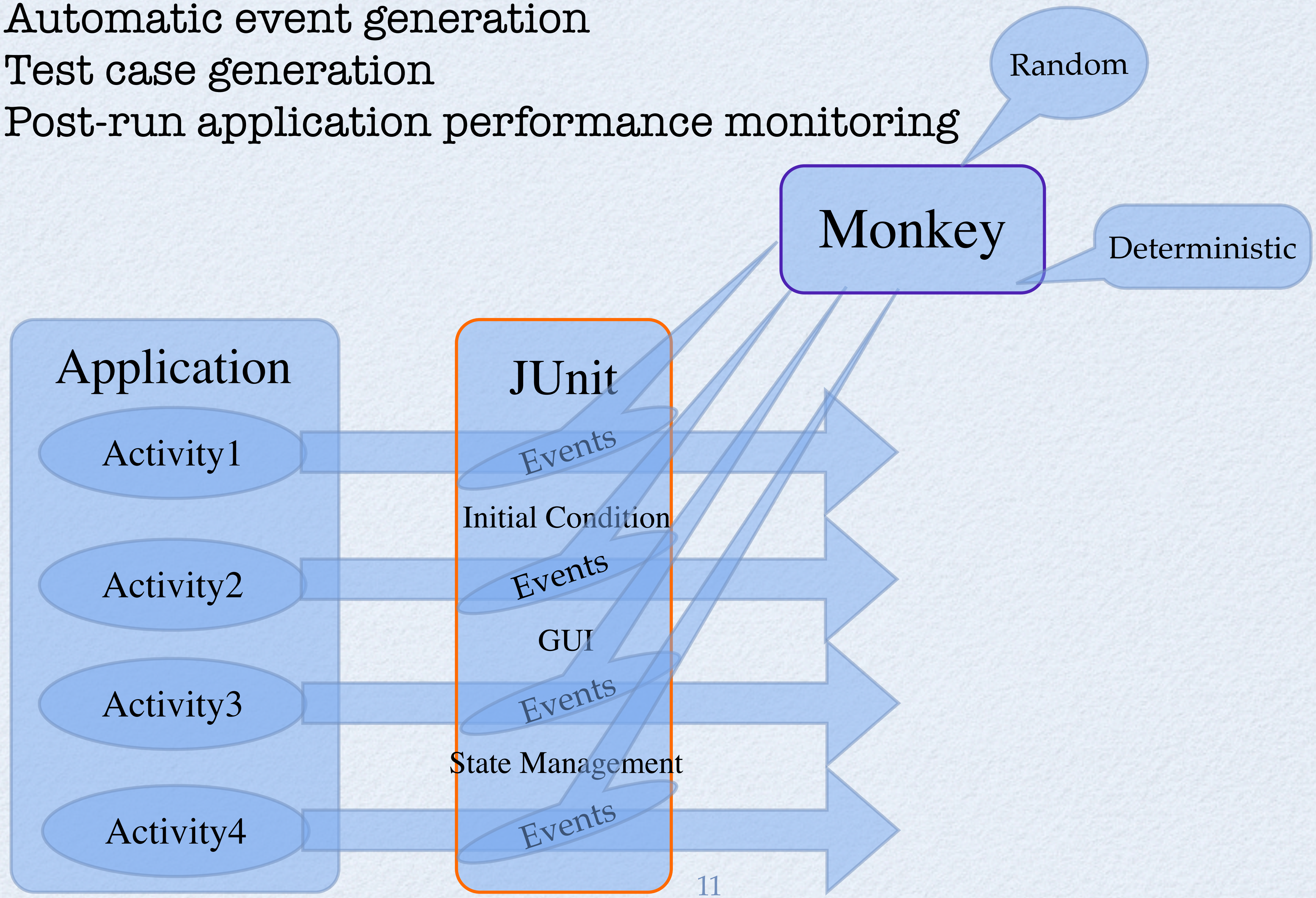
1. Automatic event generation
2. Test case generation
3. Post-run application performance monitoring



Testing Techniques

Combination of multiple methods:

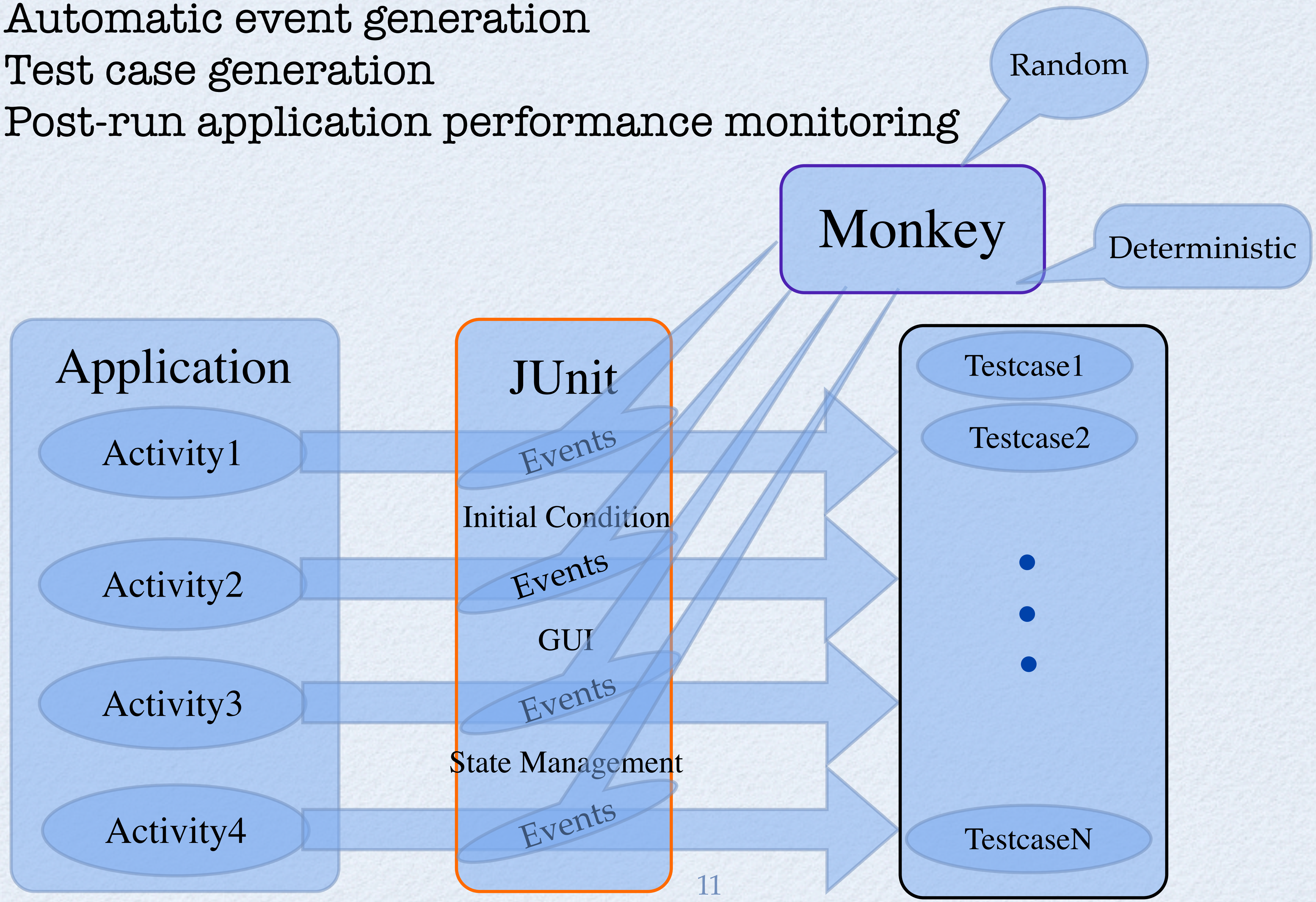
1. Automatic event generation
2. Test case generation
3. Post-run application performance monitoring



Testing Techniques

Combination of multiple methods:

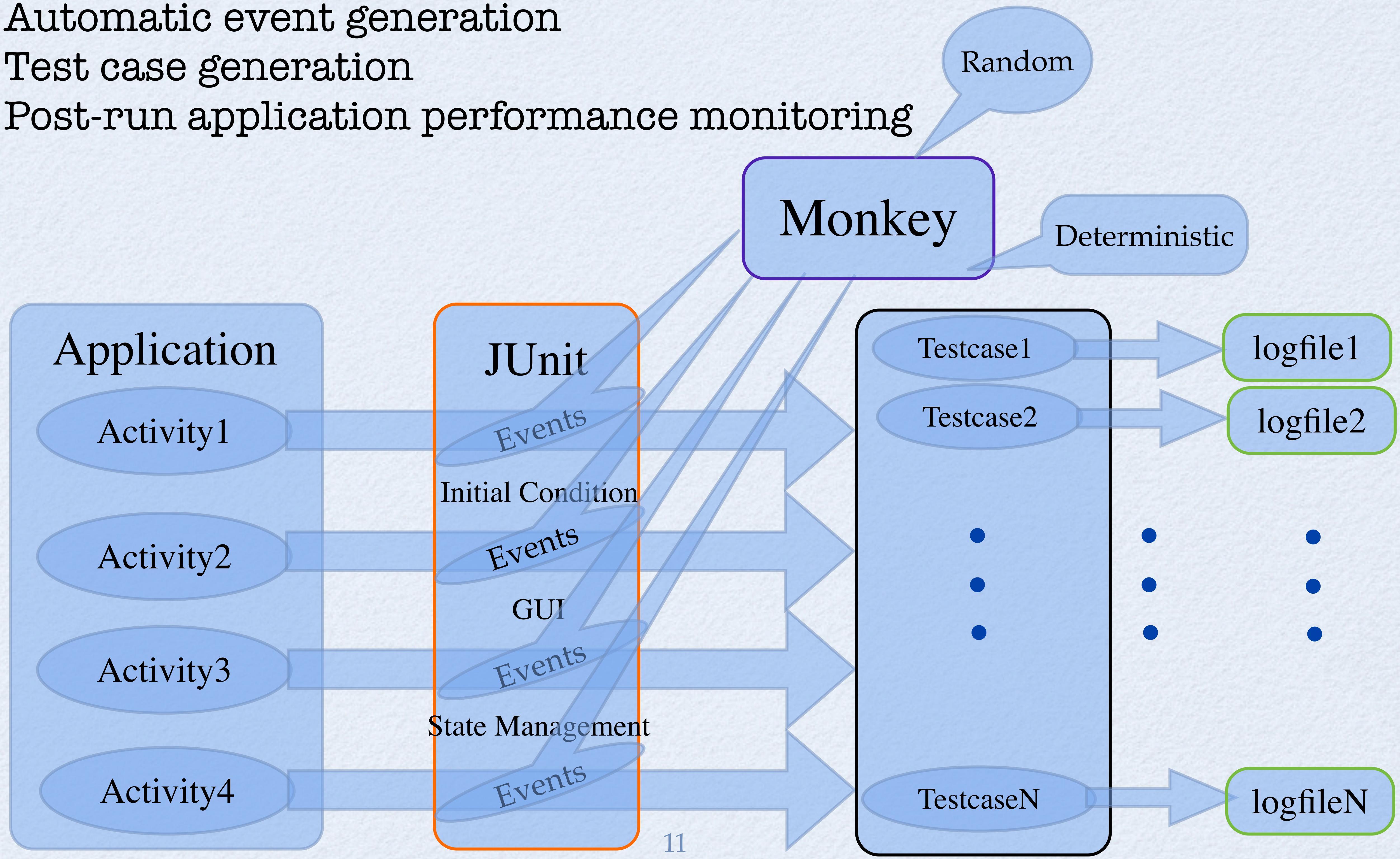
1. Automatic event generation
2. Test case generation
3. Post-run application performance monitoring



Testing Techniques

Combination of multiple methods:

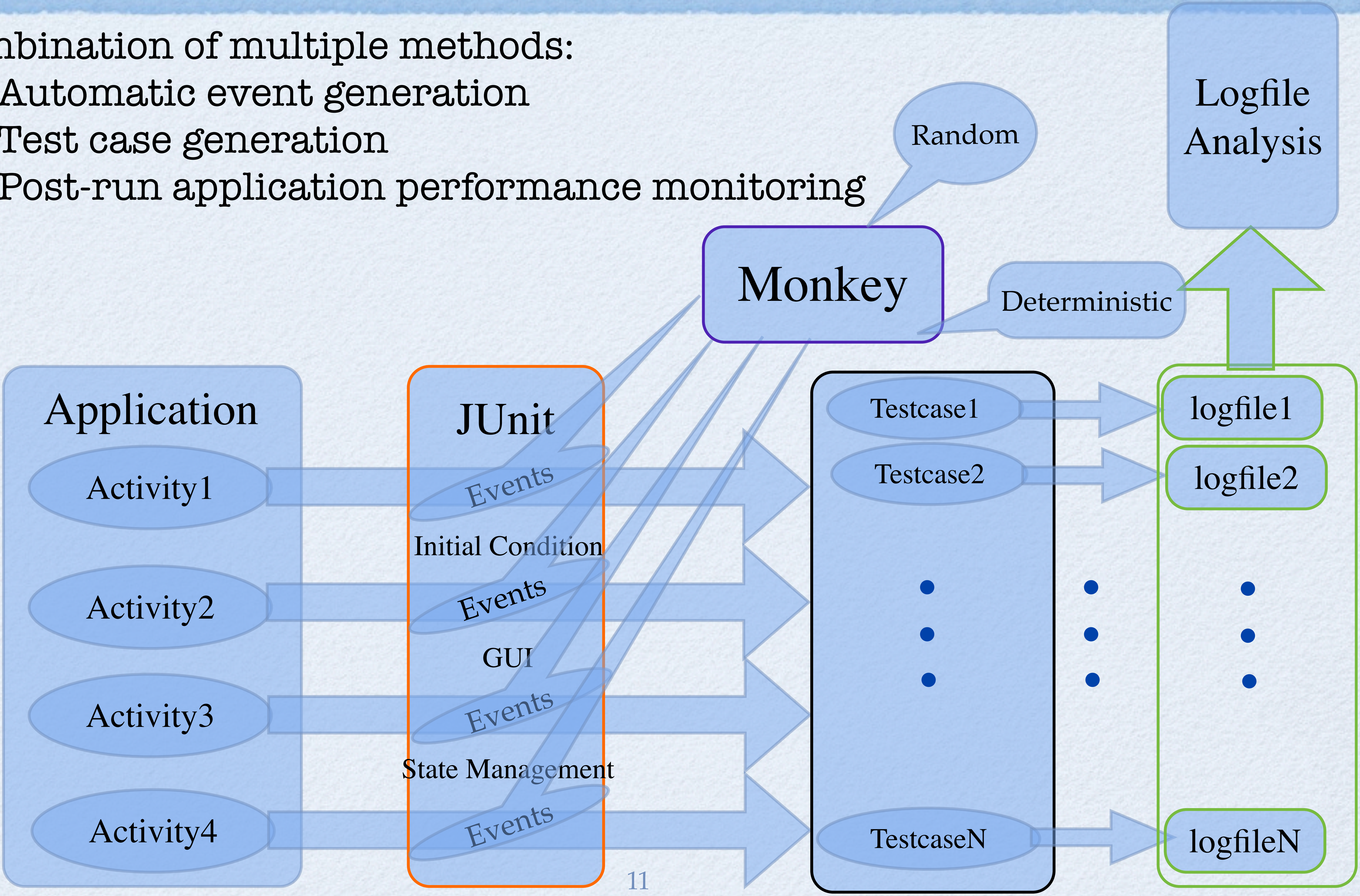
1. Automatic event generation
2. Test case generation
3. Post-run application performance monitoring



Testing Techniques

Combination of multiple methods:

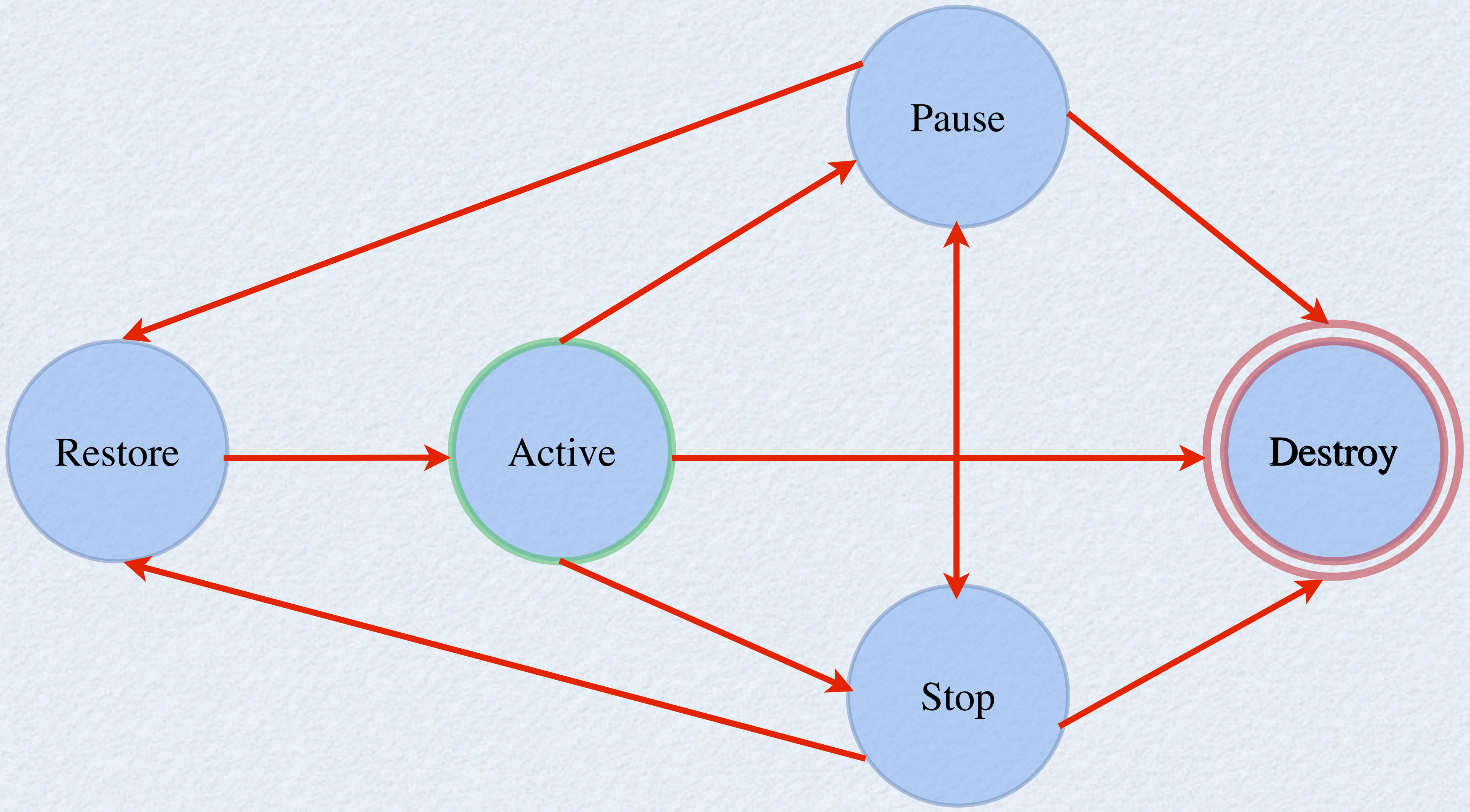
1. Automatic event generation
2. Test case generation
3. Post-run application performance monitoring



Automatic Log File Analysis

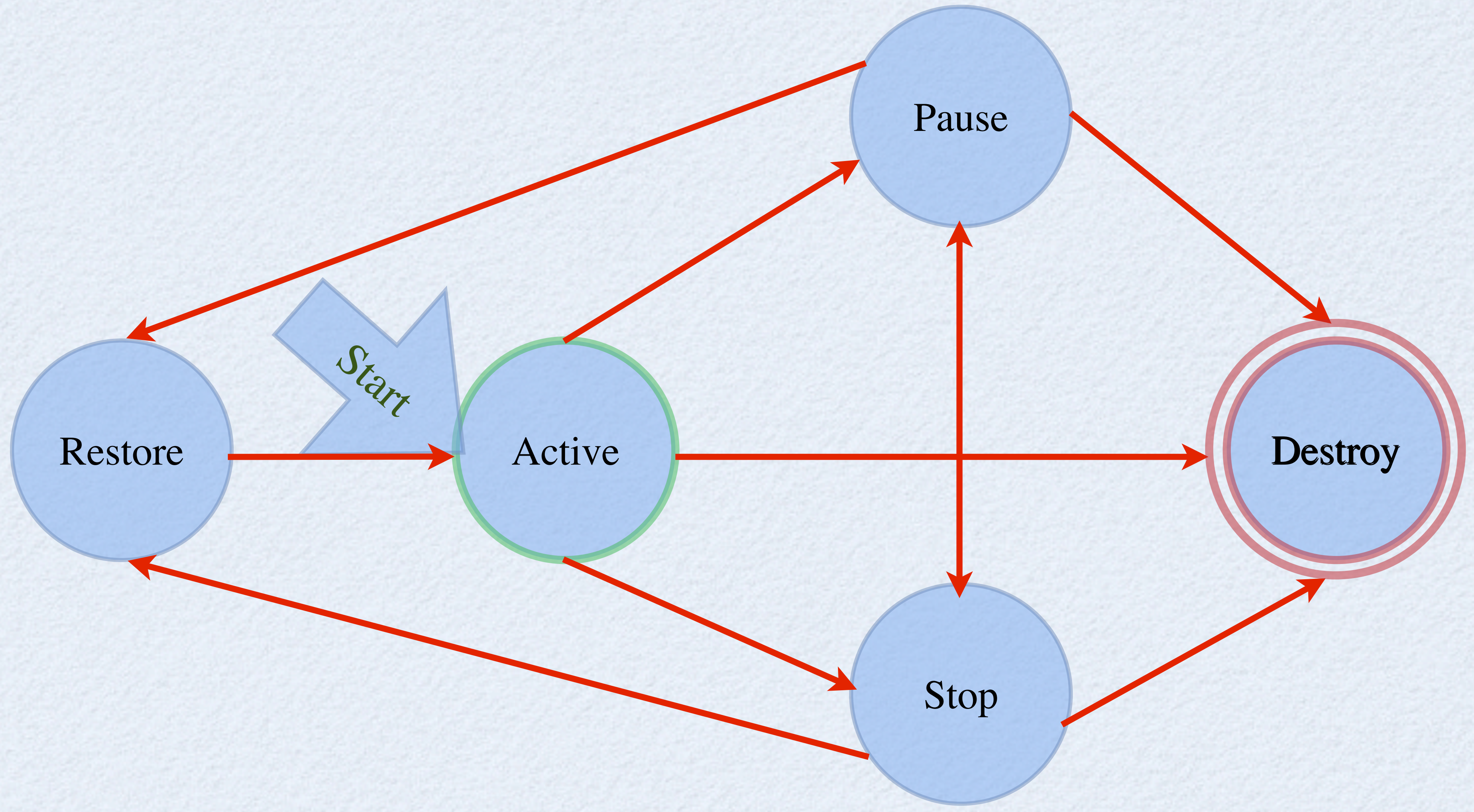
- ✦Activity and event bugs: state machine-based analysis
- ✦Runtime type errors: look for exceptions

State Machine-based Analysis



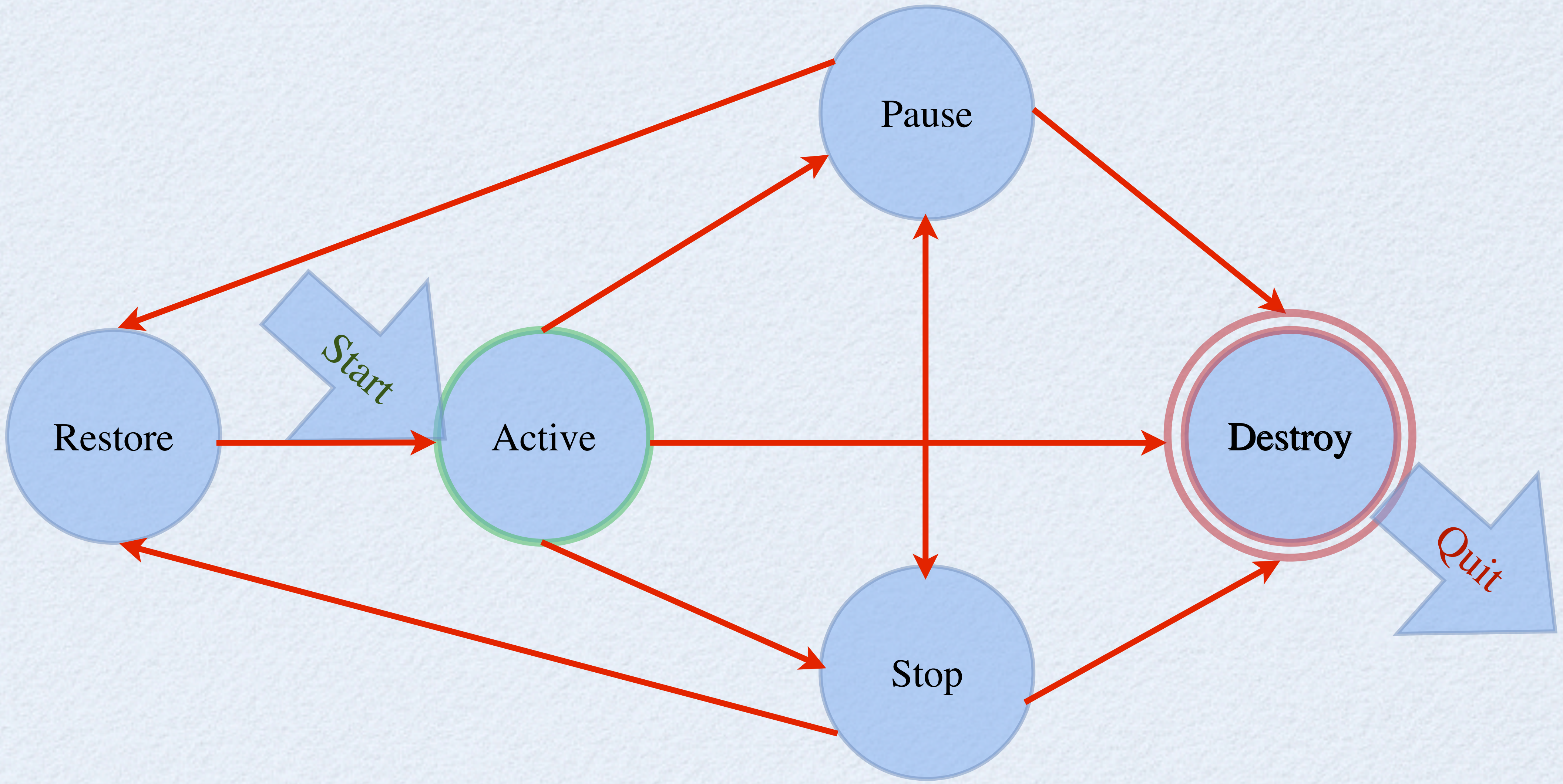
State machine used to detect activity and event bugs

State Machine-based Analysis



State machine used to detect activity and event bugs

State Machine-based Analysis



State machine used to detect activity and event bugs

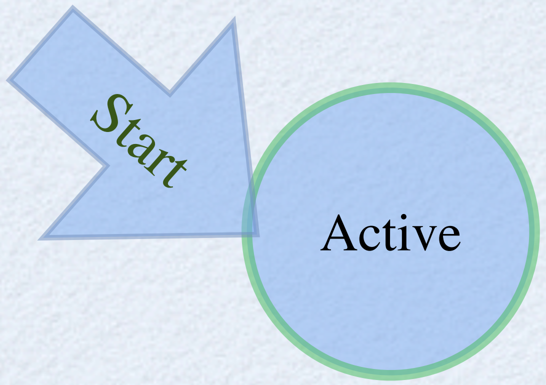
State Machine-based Analysis

Correct Pattern:

State machine used to detect activity and event bugs

State Machine-based Analysis

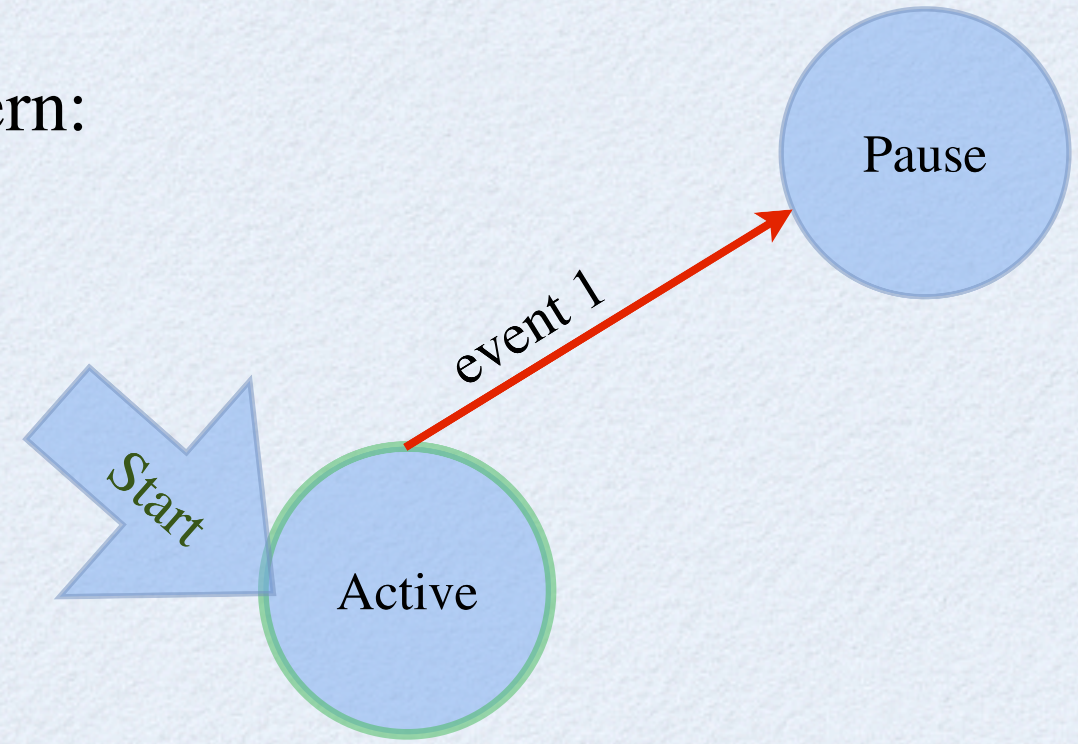
Correct Pattern:



State machine used to detect activity and event bugs

State Machine-based Analysis

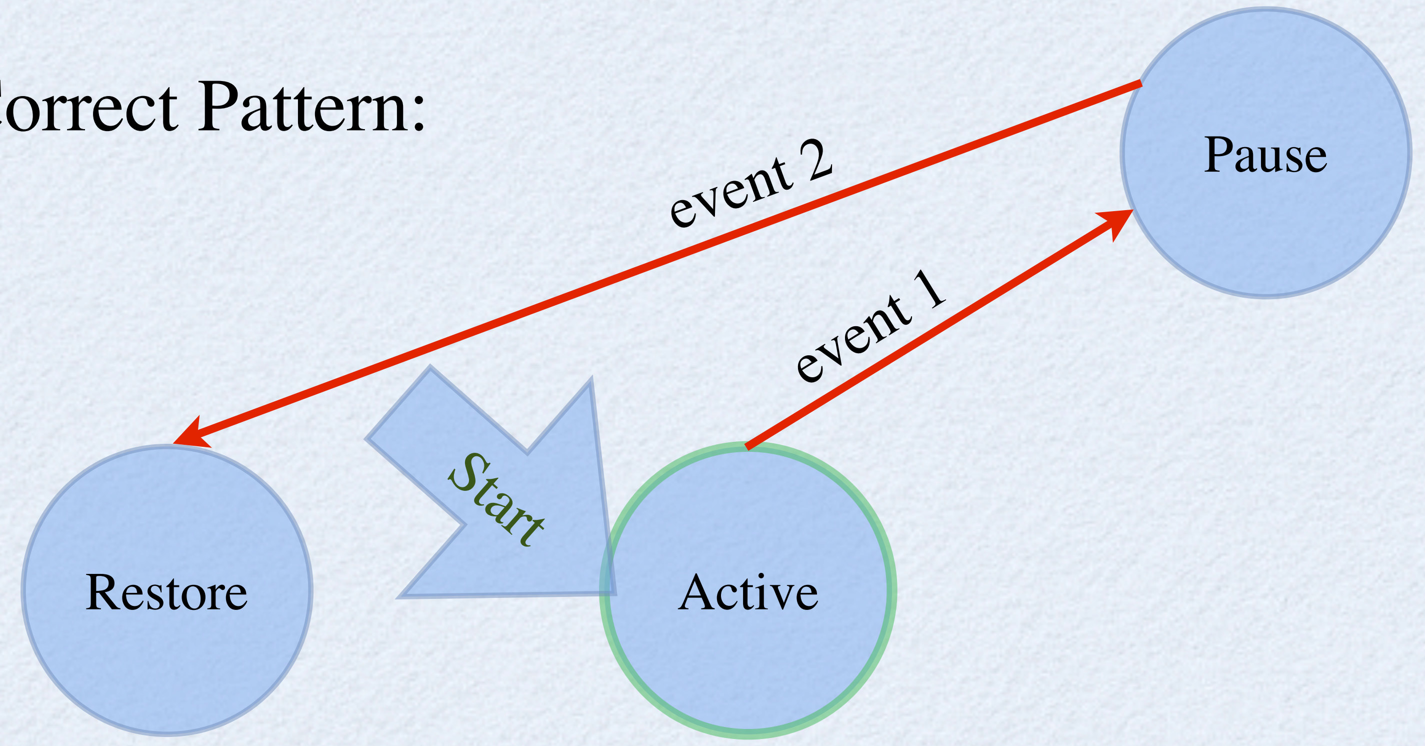
Correct Pattern:



State machine used to detect activity and event bugs

State Machine-based Analysis

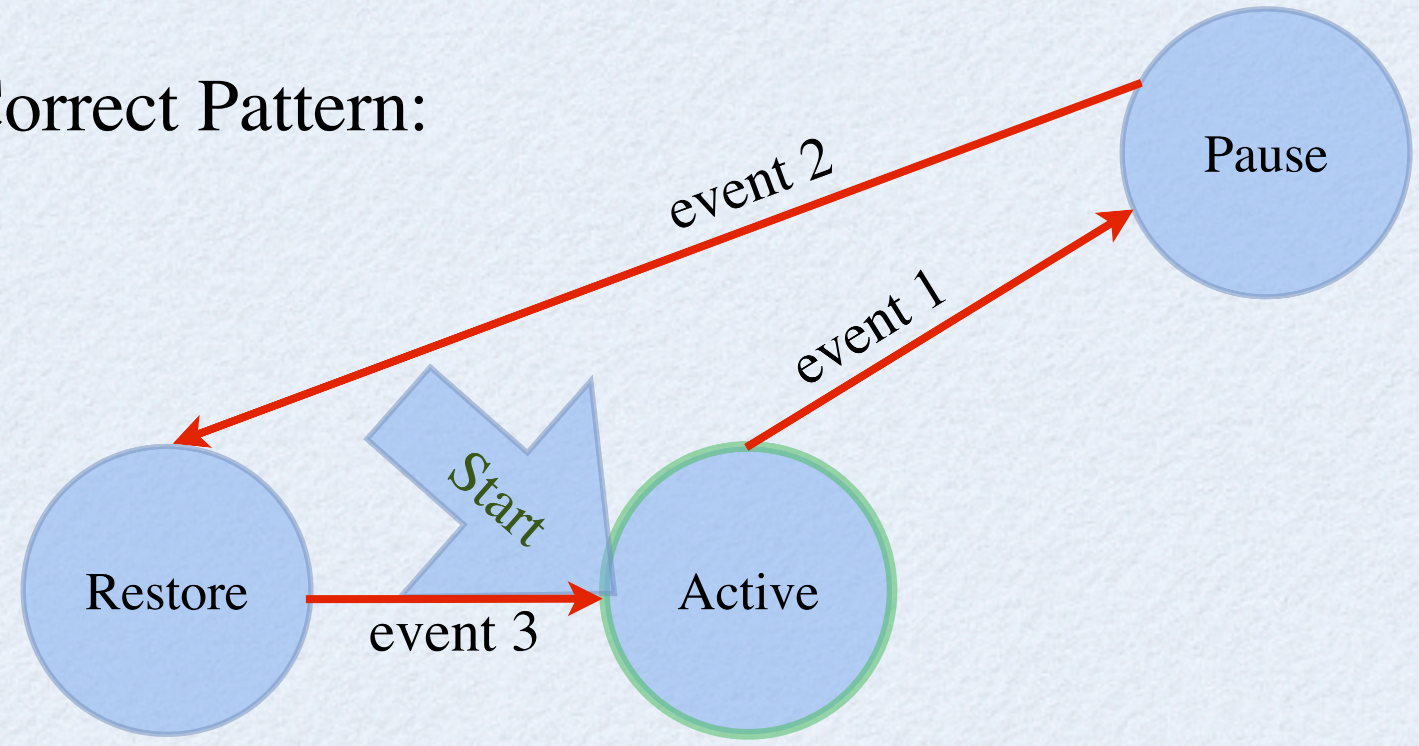
Correct Pattern:



State machine used to detect activity and event bugs

State Machine-based Analysis

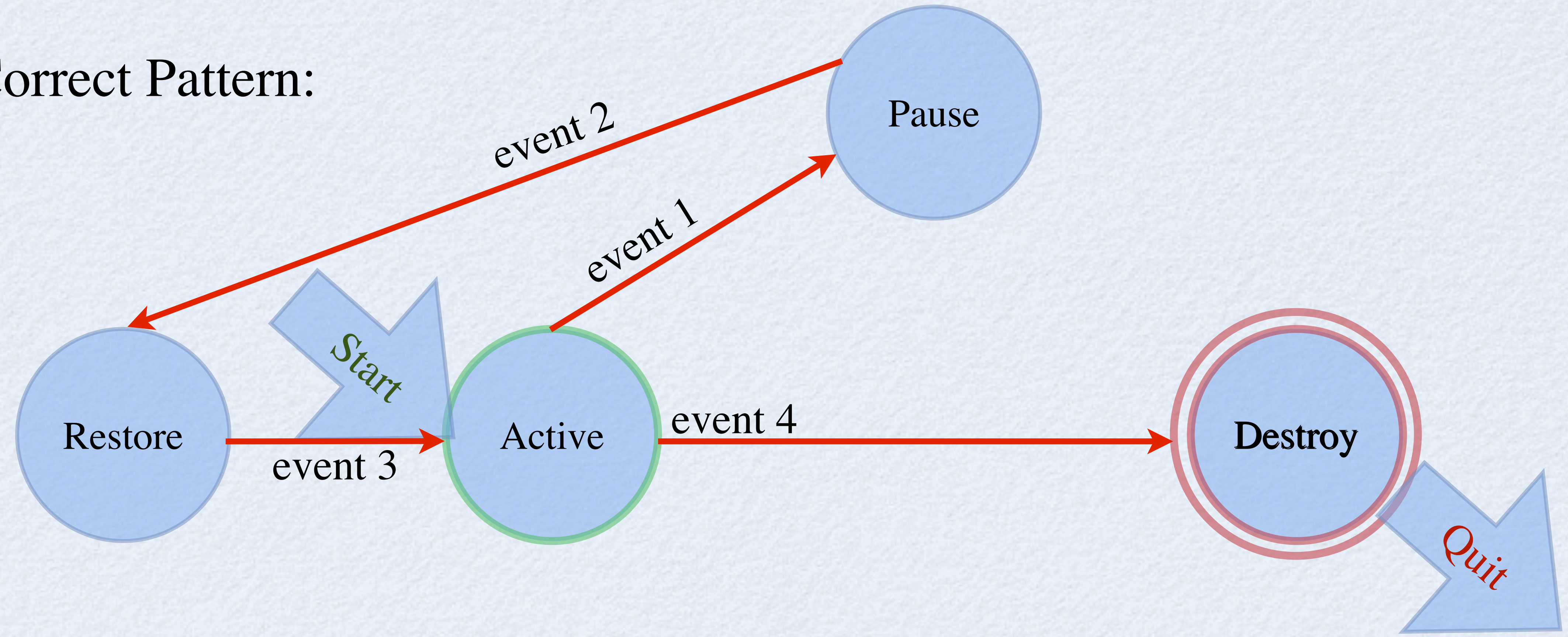
Correct Pattern:



State machine used to detect activity and event bugs

State Machine-based Analysis

Correct Pattern:



State machine used to detect activity and event bugs

State Machine-based Analysis

State machine used to detect activity and event bugs

State Machine-based Analysis

Incorrect Pattern:

State machine used to detect activity and event bugs

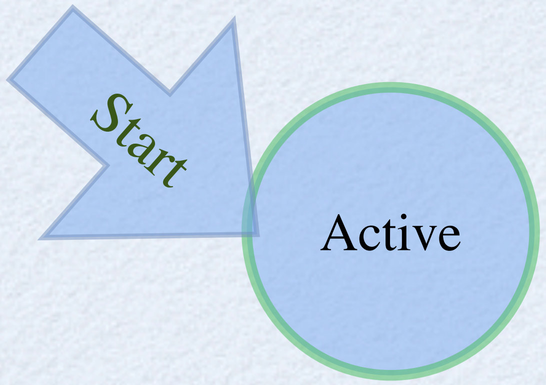
State Machine-based Analysis

Incorrect Pattern:

State machine used to detect activity and event bugs

State Machine-based Analysis

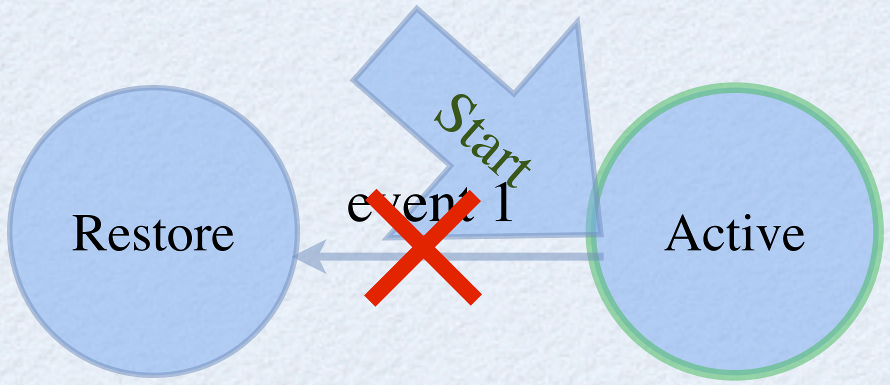
Incorrect Pattern:



State machine used to detect activity and event bugs

State Machine-based Analysis

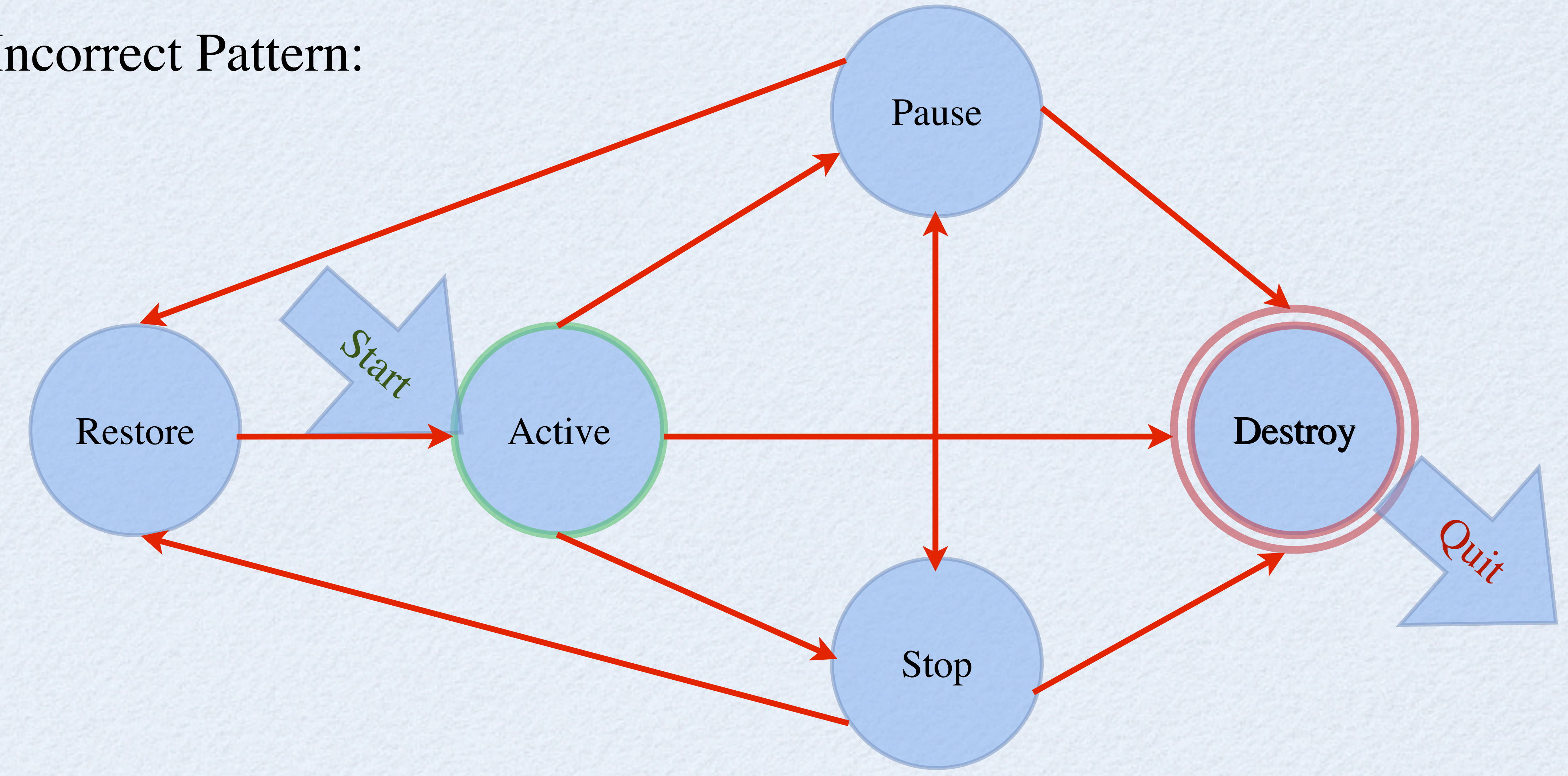
Incorrect Pattern:



State machine used to detect activity and event bugs

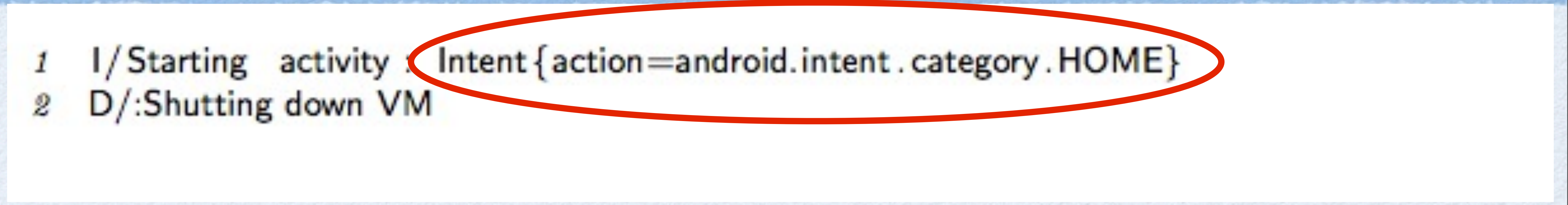
State Machine-based Analysis

Incorrect Pattern:



State machine used to detect activity and event bugs

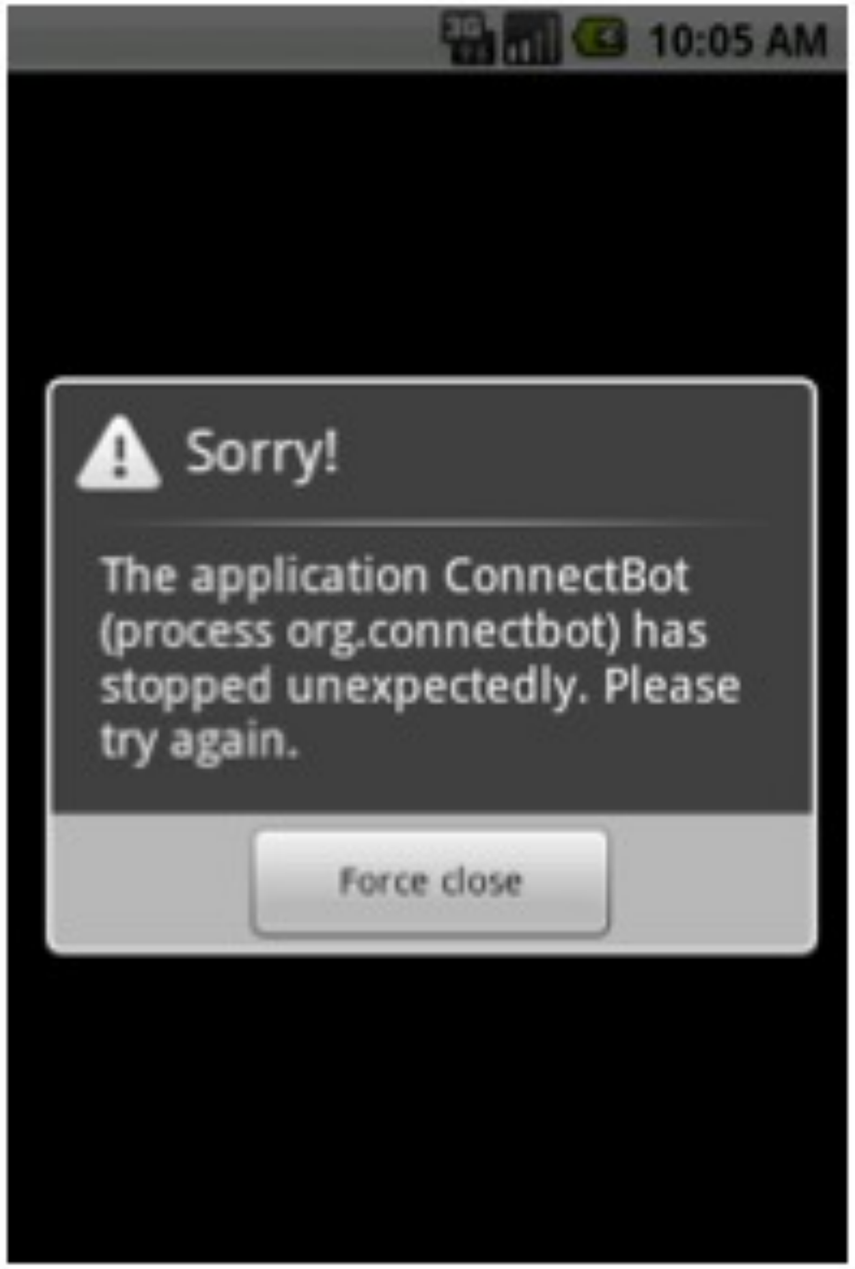
Crash Example



```
I/Starting activity : Intent{action=android.intent.category.HOME}
D/:Shutting down VM
```

Crash Example

```
I/Starting activity : Intent{action=android.intent.category.HOME}
D/:Shutting down VM
```

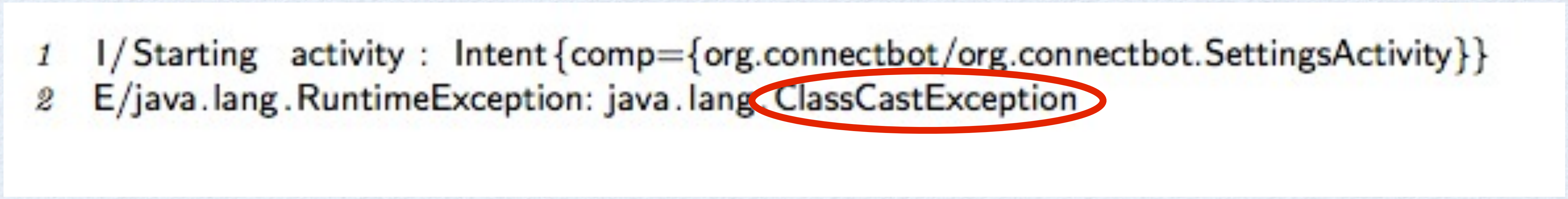


Sorry!

The application ConnectBot (process org.connectbot) has stopped unexpectedly. Please try again.

Exception Analysis

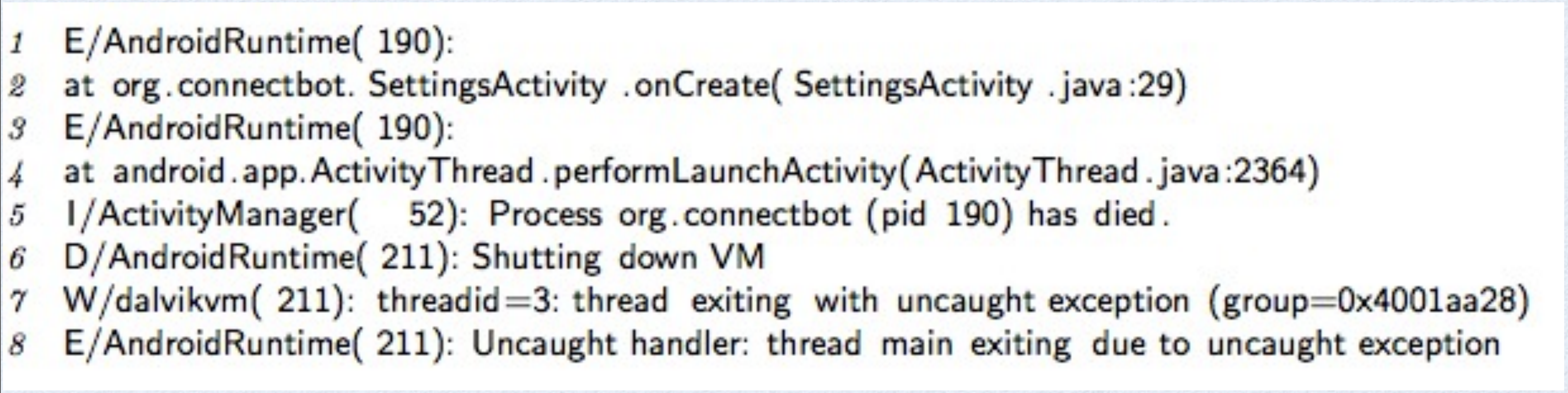
ClassCatchException in the log file



```
I/Starting activity : Intent{comp={org.connectbot/org.connectbot.SettingsActivity}}
E/java.lang.RuntimeException: java.lang.ClassCastException
```

Using Our Tool as Debugging Aid

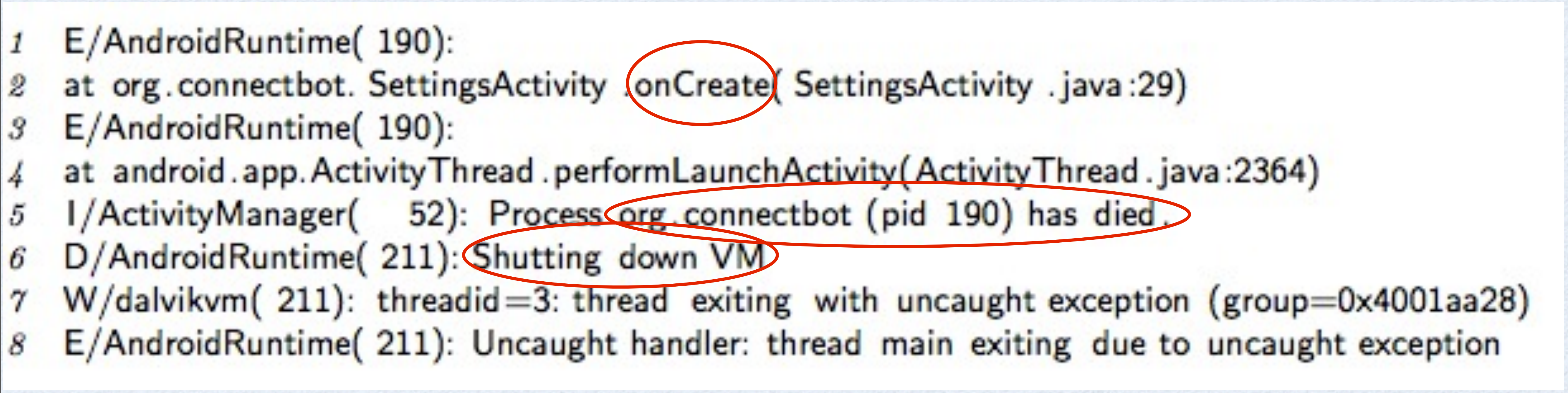
Unrolled stack helps developers reproduce bugs and pinpoint errors



```
E/AndroidRuntime( 190):
at org.connectbot. SettingsActivity .onCreate( SettingsActivity .java:29)
E/AndroidRuntime( 190):
at android.app.ActivityThread.performLaunchActivity(ActivityThread.java:2364)
I/ActivityManager(    52): Process org.connectbot (pid 190) has died.
D/AndroidRuntime( 211): Shutting down VM
W/dalvikvm( 211): threadid=3: thread exiting with uncaught exception (group=0x4001aa28)
E/AndroidRuntime( 211): Uncaught handler: thread main exiting due to uncaught exception
```

Using Our Tool as Debugging Aid

Unrolled stack helps developers reproduce bugs and pinpoint errors



```
E/AndroidRuntime( 190):
at org.connectbot. SettingsActivity .onCreate( SettingsActivity .java :29)
E/AndroidRuntime( 190):
at android.app.ActivityThread .performLaunchActivity(ActivityThread.java:2364)
I/ActivityManager(    52): Process org.connectbot (pid 190) has died.
D/AndroidRuntime( 211): Shutting down VM
W/dalvikvm( 211): threadid=3: thread exiting with uncaught exception (group=0x4001aa28)
E/AndroidRuntime( 211): Uncaught handler: thread main exiting due to uncaught exception
```

Results

Program	**Bugs reported (bugs we found)**		
	Activity	*Event*	*Type*
Skylight1	3 (3)	2 (5)	0 (0)
CMIS	0 (0)	0 (0)	0 (0)
Delicious	0 (0)	0 (0)	0 (0)
ConnectBot	2 (4)	8 (8)	2 (2)
DealDroid	1 (1)	1 (0)	0 (0)
Rokon	0 (0)	6 (6)	2 (2)
Andoku	0 (1)	0 (0)	0 (0)
Opensudoku	1 (1)	1 (2)	0 (0)
GuessTheNumber	1 (1)	1 (1)	0 (0)
MonolithAndroid	0 (0)	2 (2)	0 (0)
Total	**8 (11)**	**21 (24)**	**4 (4)**

Results

Program	Bugs reported (bugs we found)		
	Activity	*Event*	*Type*
Skylight1	3 (3)	2 (5)	0 (0)
CMIS	0 (0)	0 (0)	0 (0)
Delicious	0 (0)	0 (0)	0 (0)
ConnectBot	2 (4)	8 (8)	2 (2)
DealDroid	1 (1)	1 (0)	0 (0)
Rokon	0 (0)	6 (6)	2 (2)
Andoku	0 (1)	0 (0)	0 (0)
Opensudoku	1 (1)	1 (2)	0 (0)
GuessTheNumber	1 (1)	1 (1)	0 (0)
MonolithAndroid	0 (0)	2 (2)	0 (0)
Total	**8 (11)**	**21 (24)**	**4 (4)**

Conclusions

- ✦ Android applications prone to specific errors (activity, event)
  - ✦ We perform a bug study and categorization on 10 Android applications
- ✦ New Android-specific verification techniques needed
  - ✦ We implement a tool to detect activity/event/type bugs (Available on http://www.cs.ucr.edu/~huc)
  - ✦ Our work proved effective for detecting Android bugs
- ✦ Our approach can be extended to capture other types of bugs like API, I/O, or concurrency errors