Automatic Code Review for SmartThings Applications Using Static Analysis

by Janine Son

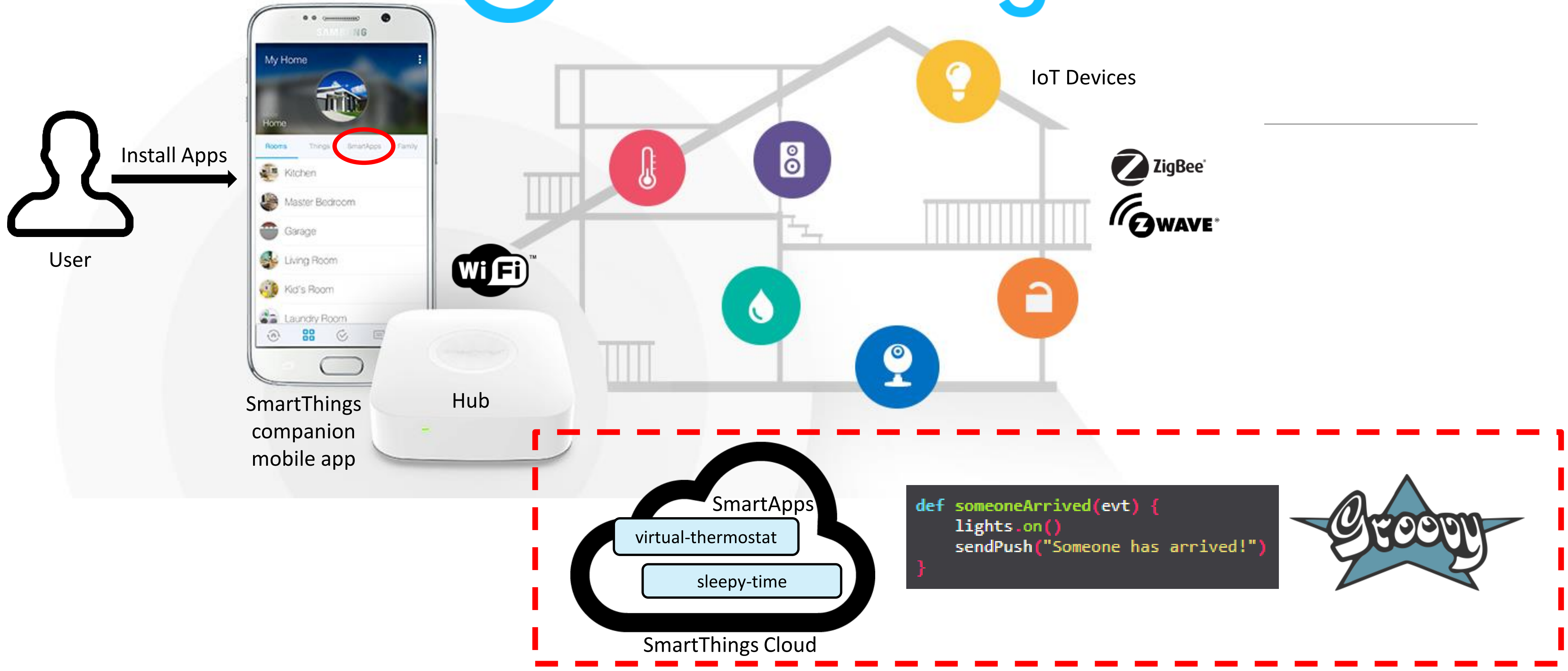
Master's Thesis Defense
Advisor: Prof. Byeong-Mo Chang

May 1, 2018
Department of Computer Science
Sookmyung Women's University, Korea

Contents

1. Introduction
2. Methodology
3. Implementation
4. Results and Discussion
5. Summary and Future Work

SmartThings®



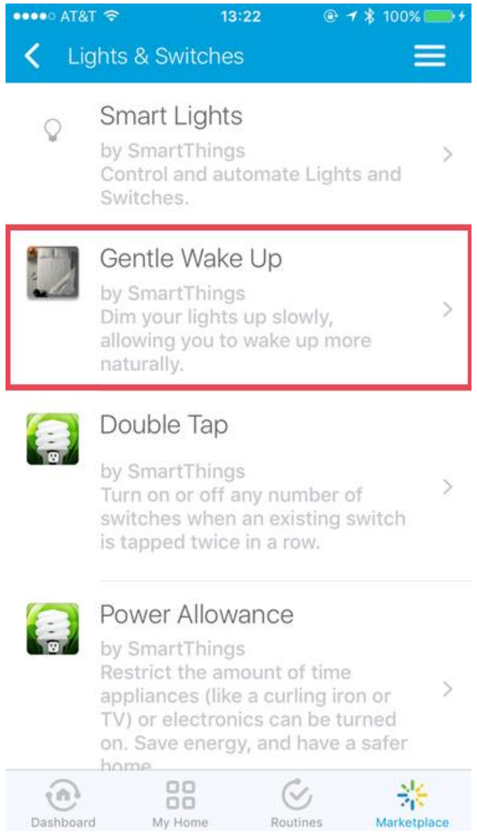
```
def someoneArrived(evt) {
    lights.on()
    sendPush("Someone has arrived!")
}
```

- Users install SmartApps through mobile devices, allowing SmartThings cloud to interact with the user's locally deployed devices
- SmartApps pair event handlers to devices, issue commands (control the IoT devices)

https://support.smartthings.com/hc/en-us/articles/205380634-Setting-up-an-account-the-Hub-and-Things

SmartApps

Official Apps



●●●●○ AT&T 13:22 100%

Lights & Switches

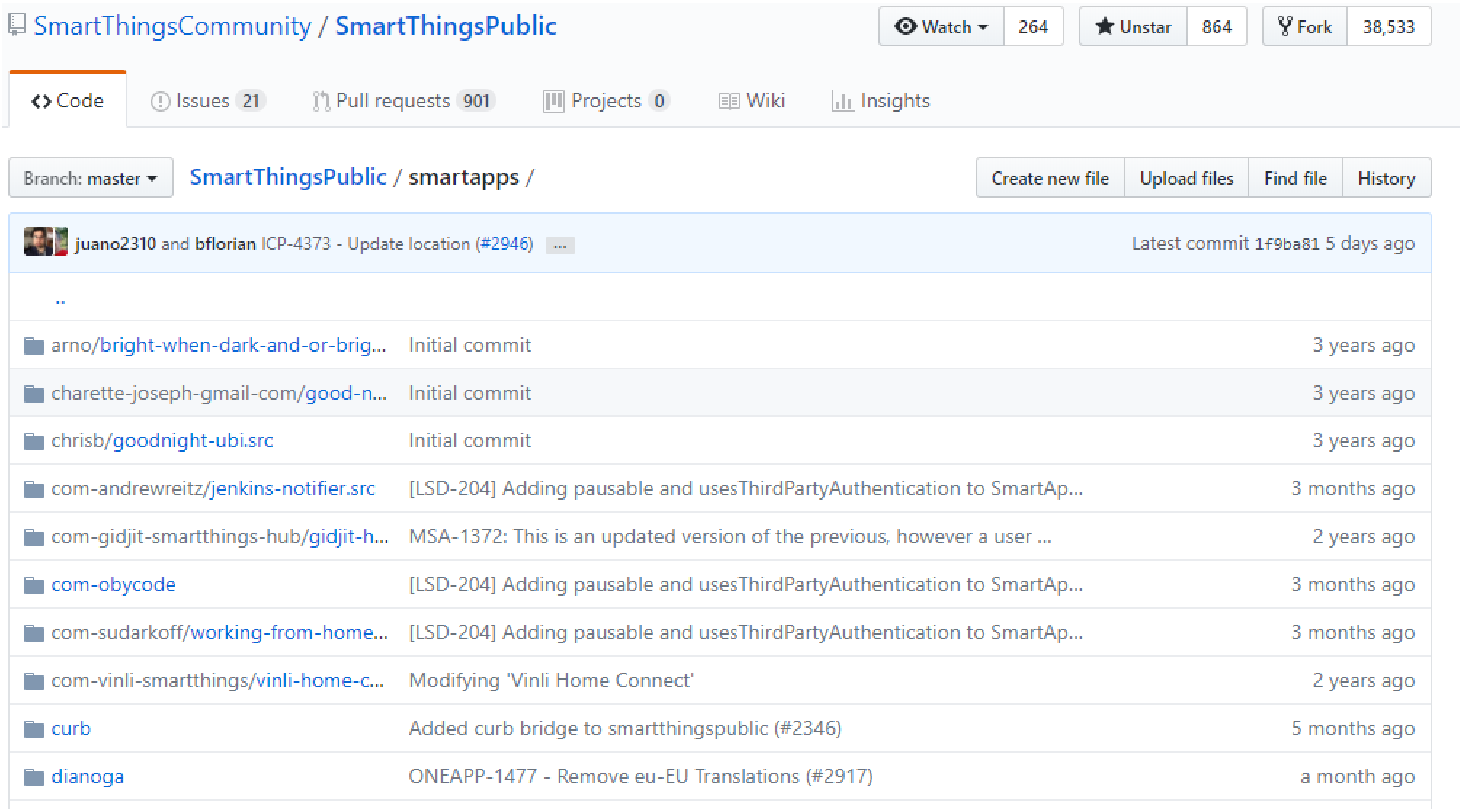
Smart Lights
by SmartThings
Control and automate Lights and Switches.

Gentle Wake Up
by SmartThings
Dim your lights up slowly, allowing you to wake up more naturally.

Double Tap
by SmartThings
Turn on or off any number of switches when an existing switch is tapped twice in a row.

Power Allowance
by SmartThings
Restrict the amount of time appliances (like a curling iron or TV) or electronics can be turned on. Save energy, and have a safer home.

Community-created Apps



SmartThingsCommunity / SmartThingsPublic

SmartThingsPublic / smartapps /

juano2310 and bflorian ICP-4373 - Update location (#2946) ... Latest commit 1f9ba81 5 days ago

..

arno/bright-when-dark-and-or-brig...	Initial commit	3 years ago
charette-joseph-gmail-com/good-n...	Initial commit	3 years ago
chrisb/goodnight-ubi.src	Initial commit	3 years ago
com-andrewreitz/jenkins-notifier.src	[LSD-204] Adding pausable and usesThirdPartyAuthentication to SmartAp...	3 months ago
com-gidjit-smartthings-hub/gidjit-h...	MSA-1372: This is an updated version of the previous, however a user ...	2 years ago
com-obycode	[LSD-204] Adding pausable and usesThirdPartyAuthentication to SmartAp...	3 months ago
com-sudarkoff/working-from-home...	[LSD-204] Adding pausable and usesThirdPartyAuthentication to SmartAp...	3 months ago
com-vinli-smartthings/vinli-home-c...	Modifying 'Vinli Home Connect'	2 years ago
curb	Added curb bridge to smartthingspublic (#2346)	5 months ago
dianoga	ONEAPP-1477 - Remove eu-EU Translations (#2917)	a month ago

https://blog.smartthings.com

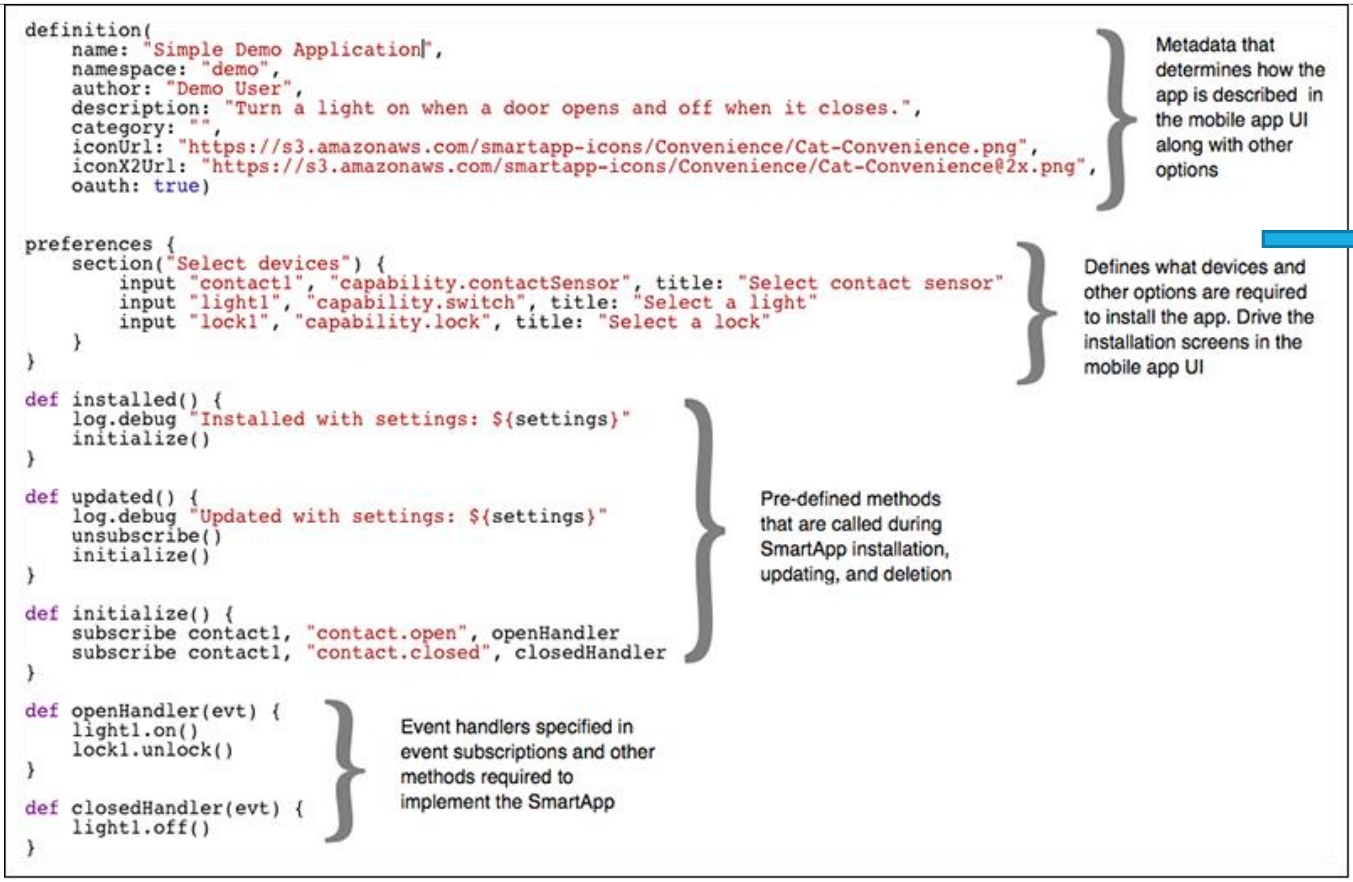
SmartApp Code Structure

Definition

Preferences

Predefined Callbacks

Event Handlers



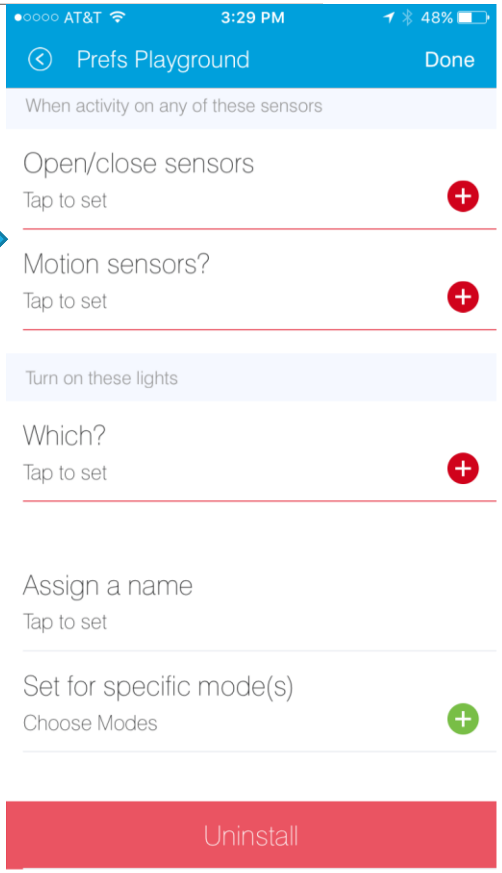
```
definition(
    name: "Simple Demo Application",
    namespace: "demo",
    author: "Demo User",
    description: "Turn a light on when a door opens and off when it closes.",
    category: "",
    iconUrl: "https://s3.amazonaws.com/smartapp-icons/Convenience/Cat-Convenience.png",
    iconX2Url: "https://s3.amazonaws.com/smartapp-icons/Convenience/Cat-Convenience@2x.png",
    oauth: true)

preferences {
    section("Select devices") {
        input "contact1", "capability.contactSensor", title: "Select contact sensor"
        input "light1", "capability.switch", title: "Select a light"
        input "lock1", "capability.lock", title: "Select a lock"
    }
}

def installed() {
    log.debug "Installed with settings: ${settings}"
    initialize()
}

def updated() {
    log.debug "Updated with settings: ${settings}"
    unsubscribe()
    initialize()
}

def initialize() {
    subscribe contact1, "contact.open", openHandler
    subscribe contact1, "contact.closed", closedHandler
}

def openHandler(evt) {
    light1.on()
    lock1.unlock()
}

def closedHandler(evt) {
    light1.off()
}
```

Metadata that determines how the app is described in the mobile app UI along with other options

Defines what devices and other options are required to install the app. Drive the installation screens in the mobile app UI

Pre-defined methods that are called during SmartApp installation, updating, and deletion

Event handlers specified in event subscriptions and other methods required to implement the SmartApp



●○○○○ AT&T 3:29 PM 48%

Prefs Playground Done

When activity on any of these sensors

Open/close sensors
Tap to set

Motion sensors?
Tap to set

Turn on these lights

Which?
Tap to set

Assign a name
Tap to set

Set for specific mode(s)
Choose Modes

Uninstall

Research Objective

- Design and develop an automatic code review tool for the evaluation of SmartApps
- Automate instead of manual code review
- Check for compliance of coding standards; ensures code is reliable, maintainable, safe

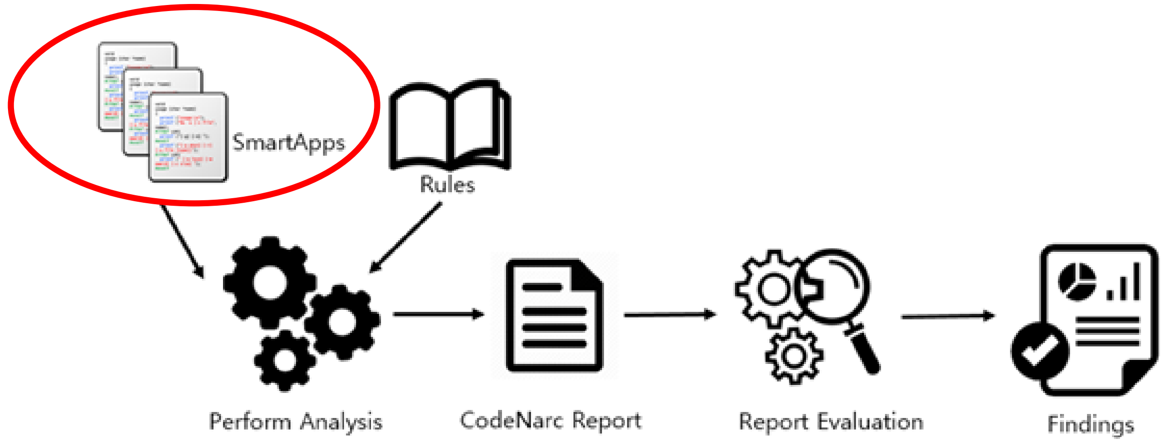
- Use static analysis approach
- Use metrics to evaluate the measurable quality attributes of SmartApps

Research Questions

- **RQ1**: What are the common violations found in SmartApps?
- **RQ2**: How do community-created SmartApps differ from official SmartApps in terms of quality?

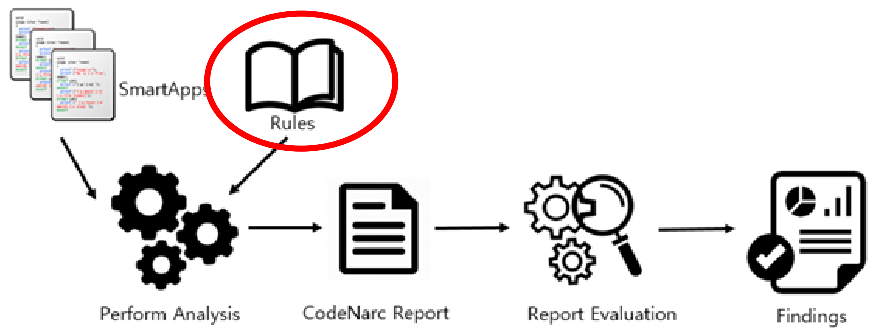
SmartApp Source Code

- 105 official and 74 community-created apps
- Source: **SmartThings Public GitHub Repository**

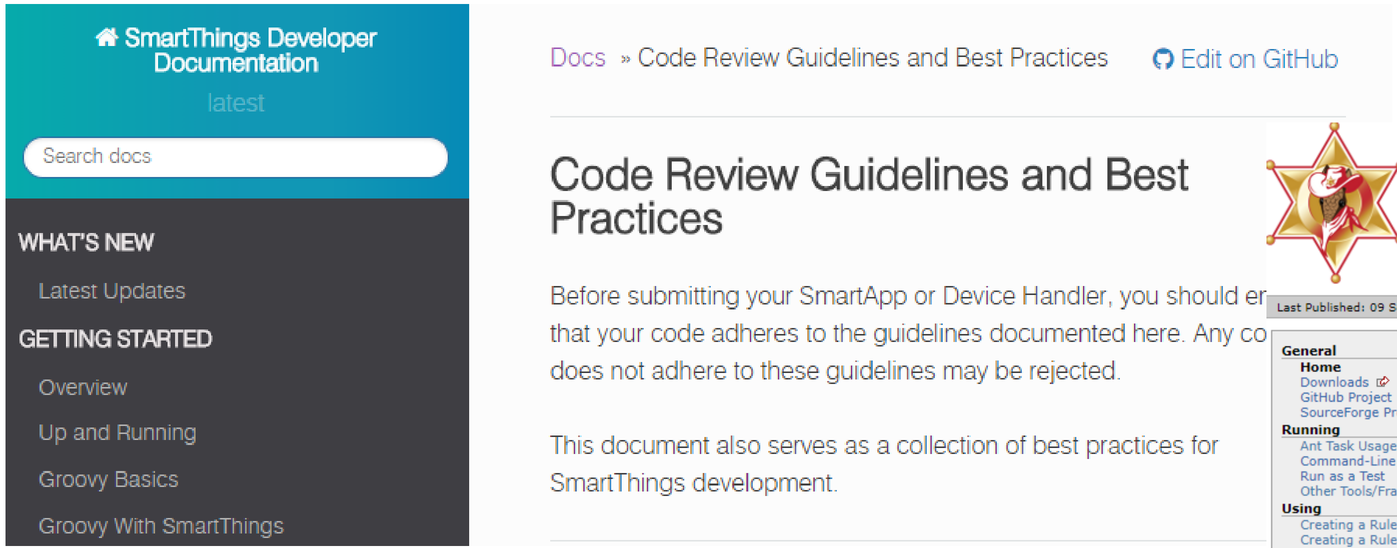


https://github.com/SmartThingsCommunity/SmartThingsPublic

Rules



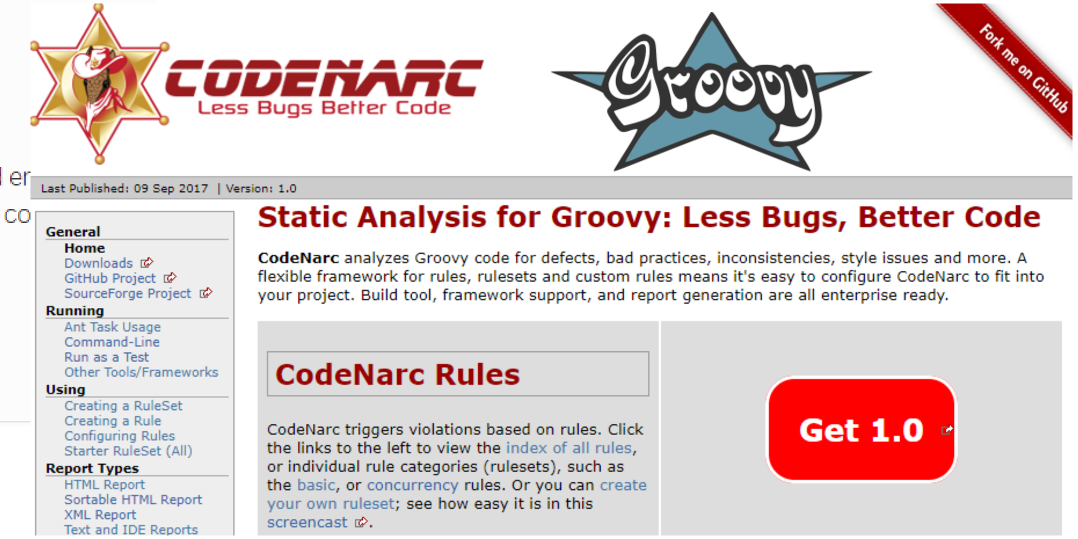
2 sources



Code Review Guidelines and Best Practices

Before submitting your SmartApp or Device Handler, you should en
that your code adheres to the guidelines documented here. Any co
does not adhere to these guidelines may be rejected.

This document also serves as a collection of best practices for SmartThings development.



Last Published: 09 Sep 2017 | Version: 1.0

Static Analysis for Groovy: Less Bugs, Better Code

CodeNarc analyzes Groovy code for defects, bad practices, inconsistencies, style issues and more. A flexible framework for rules, rulesets and custom rules means it's easy to configure CodeNarc to fit into your project. Build tool, framework support, and report generation are all enterprise ready.

CodeNarc Rules

CodeNarc triggers violations based on rules. Click the links to the left to view the index of all rules, or individual rule categories (rulesets), such as the basic, or concurrency rules. Or you can create your own ruleset; see how easy it is in this screencast.

# Total CodeNarc Rules	# Applicable Codenarc Rules	# SmartApp Rules	# Other Rules (LOC, Input, Subscription)	# Total Implemented
357	**38**	**21**	**3**	**63**

Implemented Rules

Custom Rules	Default Rules
Avoid chained runIn() calls	*Dead code*
Use consistent return values	*For loop should be while loop*
Verify array index	*Confusing ternary*
Handle null values	*Could be Elvis*
Document external HTTP requests	*If statement could be ternary*
Document exposed endpoints	*Cyclomatic complexity*
Do not hard-code SMS messages	*Nested block depth*
Missing event handler	*Assignment in conditional*
Do not use dynamic method execution	*Duplicate map key*
Subscriptions should be specific	*Empty else block*
Subscriptions should be clear	*Unused array*
Correct use of atomic state	*Comparison of two constants*
Do not use busy loop	*Constant if expression*

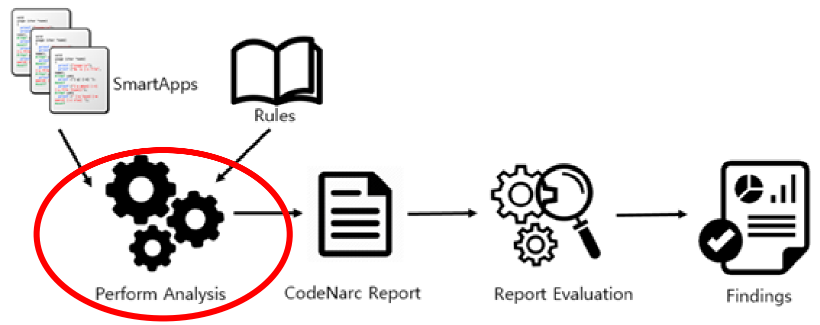
```
x ?: 'some value'
```

```
condition ? expr1 : expr2
```

26 of 63 rules

Static Analysis using CodeNarc

- CodeNarc – open source code analysis tool
- Abstract Syntax Tree (AST) traversal
- Use conditions / pattern for catching vulnerabilities



- Examples:

  1. Subscriptions should be clear:

  Visit method call 'subscribe'. If <u>attribute[1] != Constant</u> then isViolation = true

  2. Missing switch default:

  Visit switch statements. If <u>default statement == empty</u> then isViolation = true

Implementation

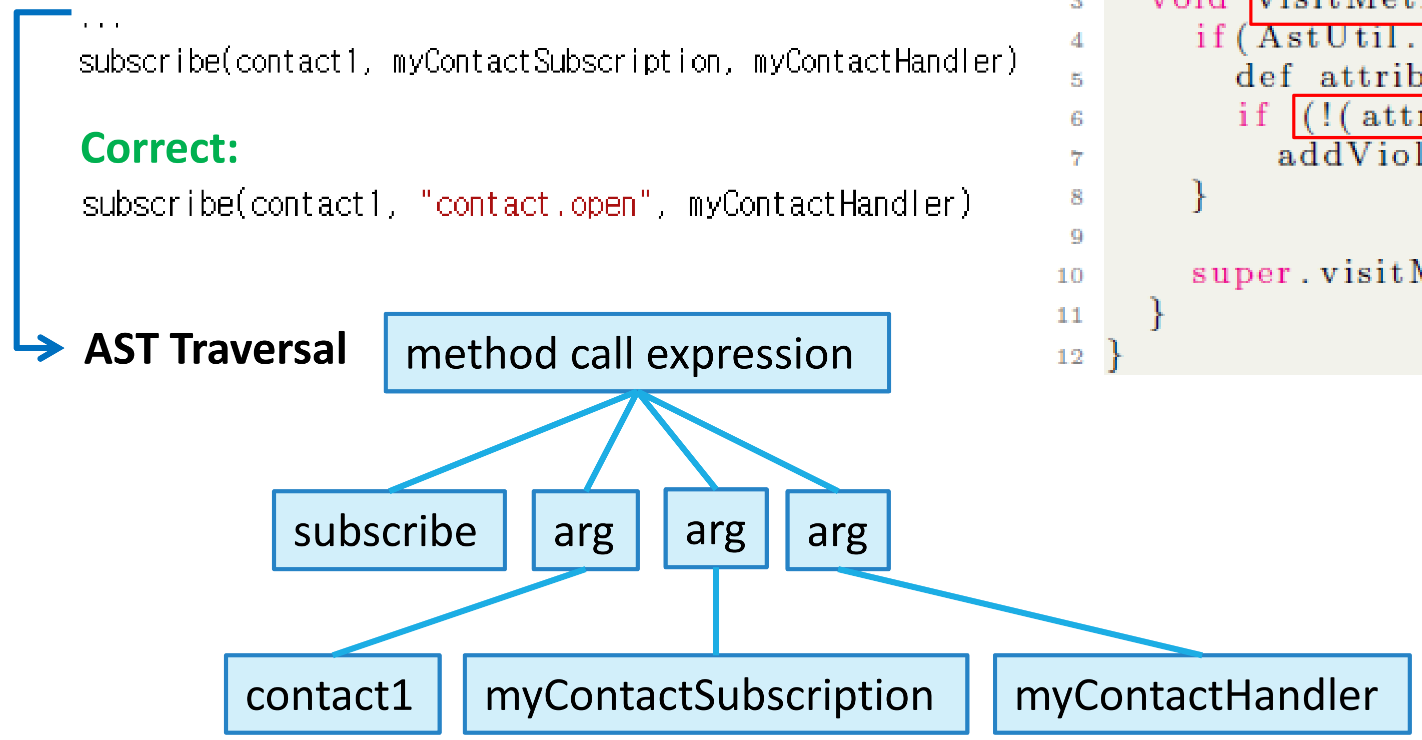
Writing NEW rule: *Subscriptions should be clear*

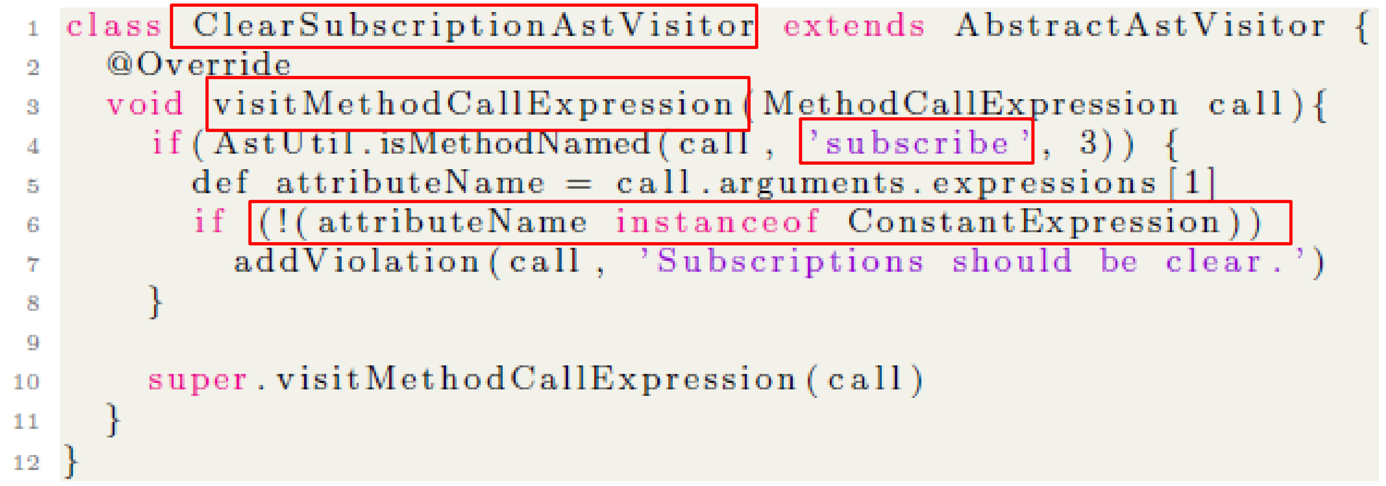
Violation:

```
def myContactSubscription = "contact.open"
...
subscribe(contact1, myContactSubscription, myContactHandler)
```

Correct:

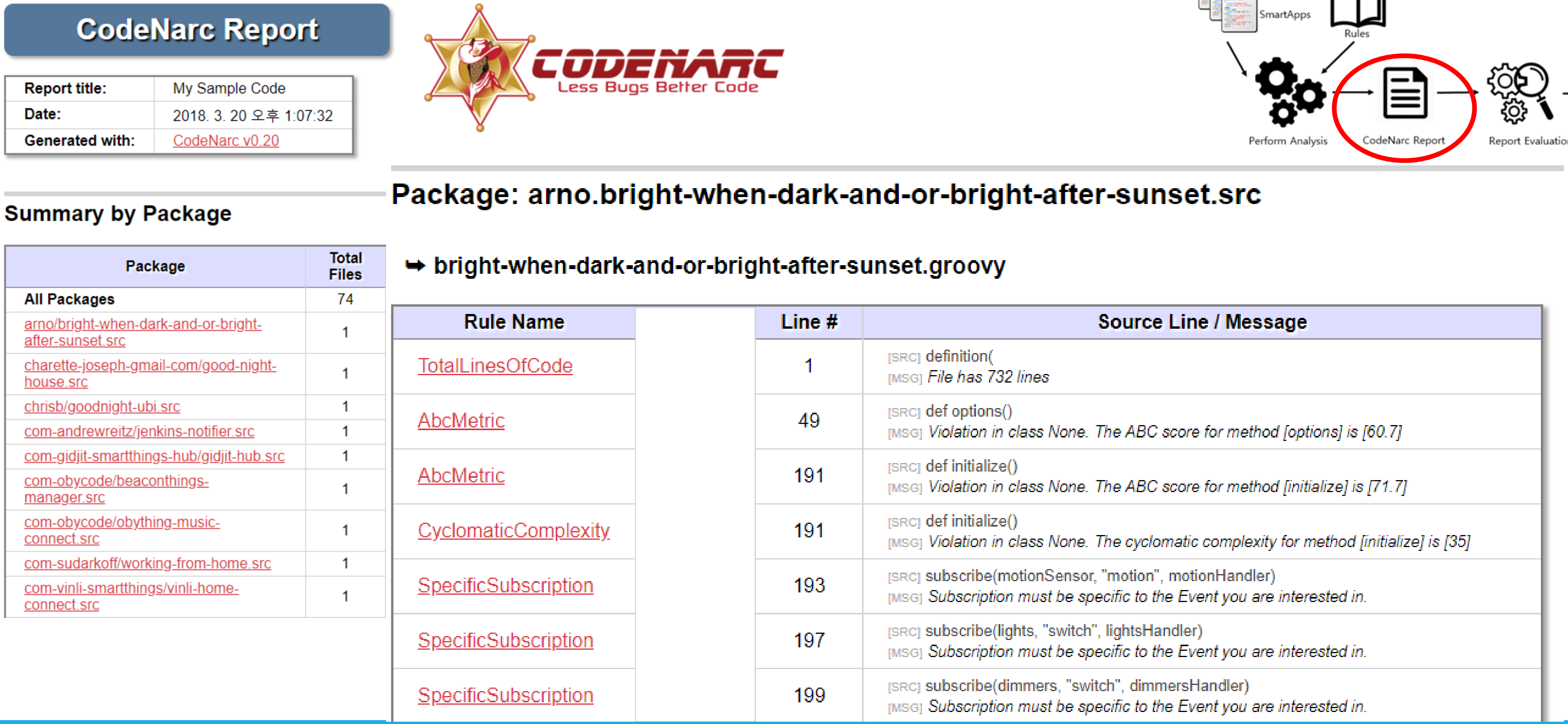
```
subscribe(contact1, "contact.open", myContactHandler)
```





```
class ClearSubscriptionAstVisitor extends AbstractAstVisitor {
  @Override
  void visitMethodCallExpression(MethodCallExpression call){
    if(AstUtil.isMethodNamed(call, 'subscribe', 3)) {
      def attributeName = call.arguments.expressions[1]
      if (!(attributeName instanceof ConstantExpression))
        addViolation(call, 'Subscriptions should be clear.')
    }

    super.visitMethodCallExpression(call)
  }
}
```

CodeNarc Report



CodeNarc Report

Report title:	My Sample Code
Date:	2018. 3. 20 오후 1:07:32
Generated with:	CodeNarc v0.20

CODENARC
Less Bugs Better Code

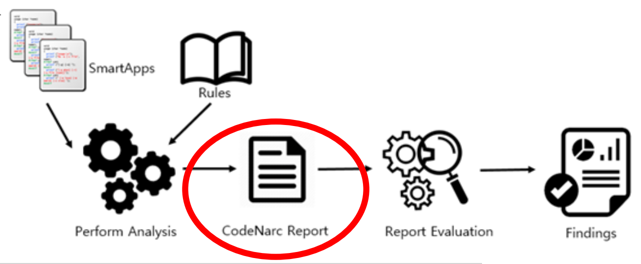
Summary by Package

Package	Total Files
All Packages	74
arno/bright-when-dark-and-or-bright-after-sunset.src	1
charette-joseph-gmail-com/good-night-house.src	1
chrisb/goodnight-ubi.src	1
com-andrewreitz/jenkins-notifier.src	1
com-gidjit-smartthings-hub/gidjit-hub.src	1
com-obycode/beaconthings-manager.src	1
com-obycode/obything-music-connect.src	1
com-sudarkoff/working-from-home.src	1
com-vinli-smartthings/vinli-home-connect.src	1

Package: arno.bright-when-dark-and-or-bright-after-sunset.src

➥ bright-when-dark-and-or-bright-after-sunset.groovy

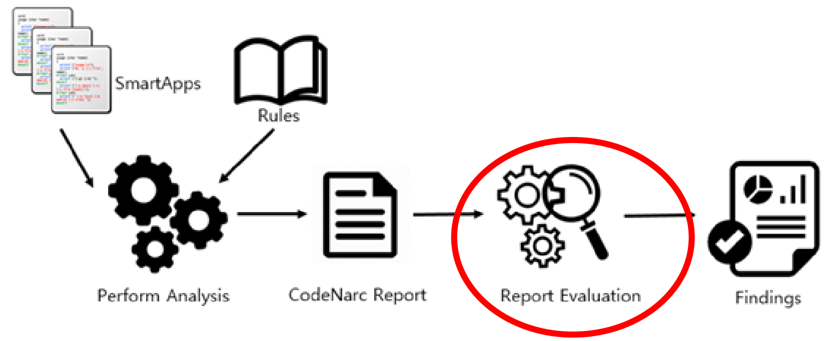
Rule Name		Line #	Source Line / Message
TotalLinesOfCode		1	[SRC] definition([MSG] *File has 732 lines*
AbcMetric		49	[SRC] def options() [MSG] *Violation in class None. The ABC score for method [options] is [60.7]*
AbcMetric		191	[SRC] def initialize() [MSG] *Violation in class None. The ABC score for method [initialize] is [71.7]*
CyclomaticComplexity		191	[SRC] def initialize() [MSG] *Violation in class None. The cyclomatic complexity for method [initialize] is [35]*
SpecificSubscription		193	[SRC] subscribe(motionSensor, "motion", motionHandler) [MSG] *Subscription must be specific to the Event you are interested in.*
SpecificSubscription		197	[SRC] subscribe(lights, "switch", lightsHandler) [MSG] *Subscription must be specific to the Event you are interested in.*
SpecificSubscription		199	[SRC] subscribe(dimmers, "switch", dimmersHandler) [MSG] *Subscription must be specific to the Event you are interested in.*



Evaluation Metrics

Quality Attributes (from ISO SQuaRE)

- *Reliability* – evaluate the frequency of faults
- *Maintainability* – evaluate the easiness of identifying styles, structure, behavior, and parts for maintenance
- *Security* – evaluate the possibility of vulnerabilities and attacks



Evaluation Tool

- Input: CodeNarc HTML report
- Parse HTML and calculate code defect rate

$$code\ defect\ density = \frac{defects}{lines\ of\ code} \times 1000$$

Rules Associated with Quality Attributes

Does the **rule / analysis tool metric** address the **quality attribute goal** ? MATCH

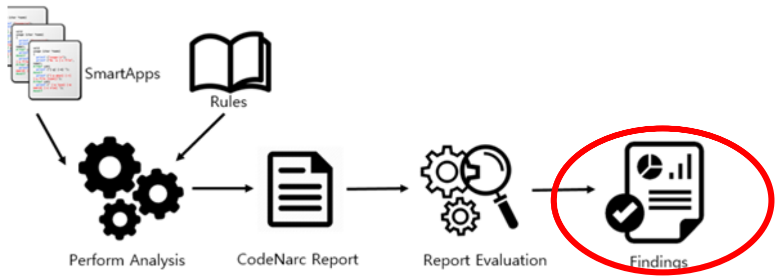
Custom Rules	Quality Attribute
Avoid chained runIn() calls	Reliability
Use consistent return values	Reliability
Verify array index	Reliability
Handle null values	Reliability
Document external HTTP requests	Security
Document exposed endpoints	Security
Do not hard-code SMS messages	Security
Missing event handler	Reliability
Do not use dynamic method execution	Security
Subscriptions should be specific	Security
Subscriptions should be clear	Security
Correct use of atomic state	Reliability
Do not use busy loop	Reliability

Default Rules	Quality Attribute
Dead code	Reliability
For loop should be while loop	Maintainability
Confusing ternary	Maintainability
Could be Elvis	Maintainability
If statement could be ternary	Maintainability
Cyclomatic complexity	Maintainability
Nested block depth	Maintainability
Assignment in conditional	Reliability
Duplicate map key	Reliability
Empty else block	Reliability
Unused array	Reliability
Comparison of two constants	Reliability
Constant if expression	Reliability

Evaluation Report

```
FILENAME :  routine-director.groovy
rule name : SpecificSubscription line : 103
source line/ message : [SRC]subscribe(people, "presence", presence)
[MSG]Subscription must be specific to the Event you are interested in.
rule name : AvoidRecurringShortSchedules line : 118
source line/ message : [SRC]runIn(60,"setSunrise")
[MSG]Avoid recurring short schedules unless there is a good reason for it.
rule name : AvoidRecurringShortSchedules line : 122
source line/ message : [SRC]runIn(60,"setSunset")
[MSG]Avoid recurring short schedules unless there is a good reason for it.
---Defect Density Metrics (KLOC)---
Reliability - 7.66
Security - 3.83
Maintainability - 0.0
Total Defect Density - 11.49
---Breakdown of Violations and Other Metrics---
Lines of Code : 261
No. of Device Input : 1
No. of Subscriptions : 1
AvoidRecurringShortSchedules : 2
SpecificSubscription : 1
Total Violations : 3
```

↑ Code defect density ↓ Quality of code

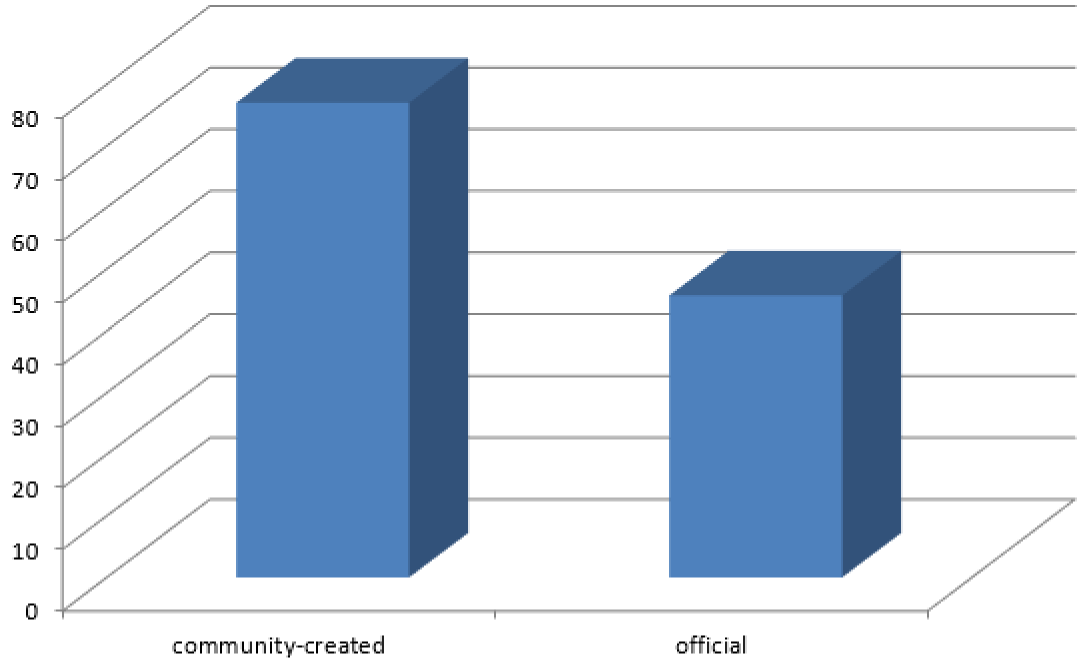


Summary

```
Total SmartApps Analyzed : 74
Total SmartApps with Violations : 57
Defect Density Mean : 31.767543859649123
---Most Common Violations---
SpecificSubscription:31
DocumentExposedEndpoints:18
DocumentExternalHTTPRequests:15
InvertedIfElse:15
CyclomaticComplexity:13
MethodCount:8
CouldBeElvis:8
MissingSwitchDefault:7
EmptyMethod:7
AtomicStateUsage:6
```

Results and Discussion

SmartApp Type	# Total Analyzed	# With Violations (%)	# Violated Rules
Official	105	48 (45.7%)	25
Community-created	74	57 (77%)	25



Top 10 Common Violations

- **RQ1**: What are the common violations found in SmartApps?

Rule	Quality Attribute	% Community-created	% Official
Subscriptions should be specific	Security	42	23
Document exposed endpoints	Security	24	10
Inverted if-else	Maintainability	20	11
Document external HTTP requests	Security	20	10
Cyclomatic complexity	Maintainability	18	4
Method count	Reliability	11	10
Could be Elvis	Maintainability	11	8
Use consistent return values	Reliability	3	10
Missing switch default	Reliability	9	5
Empty method	Reliability	9	5

Top 1: *Subscriptions should be specific*

Security - Ensure the validity of code to be executed for a particular purpose.

- The best practice is to create subscriptions specific to the Event you are interested in
- Example: Broad subscription to ‘lock’ attribute will trigger the handler when device status changes to lock and unlock. Handler will be executed on unintended device status ‘lock.unlock’.

✗

```
preferences {
    section("Select lock/s...") {
        input "lock1","capability.lock", multiple: true
    }
}

def installed()
{
    subscribe(lock1, "lock", lockHandler)
}

def lockHandler(evt)
{
    if (evt.value == "lock")
        sendMessage("Doors locked")
}
```

✓

```
preferences {
    section("Select lock/s...") {
        input "lock1","capability.lock", multiple: true
    }
}

def installed()
{
    subscribe(lock1, "lock.lock", lockHandler)
}

def lockHandler(evt)
{
    sendMessage("Doors locked")
}
```

Top 2: *Document exposed endpoints*

Security

```
mappings {
    path("/foo") {
        action: [
            GET: "getFoo",
            PUT: "putFoo",
            POST: "postFoo",
            DELETE: "deleteFoo"
        ]
    }
    path("/bar") {
        action: [
            GET: "getBar"
        ]
    }
}
```

Top 3: *Inverted if-else*

Maintainability

✗

```
if (!x) {
    false
} else {
    true
}
```

✓

```
if (x) {
    false
} else {
    true
}
```

Top 4: *Document external HTTP requests*

Security

```
def params = [
    uri: "http://httpbin.org",
    path: "/get"
]

try {
    httpGet(params) { resp ->
        resp.headers.each {
            log.debug "${it.name} : ${it.value}"
        }
        log.debug "response contentType: ${resp.contentType}"
        log.debug "response data: ${resp.data}"
    }
} catch (e) {
    log.error "something went wrong: $e"
}
```

Top 5: *Cyclomatic complexity*

Maintainability

maxMethodComplexity = 5

```
def myMethod() {
    a && b && c && d && e && f
}
```

```
def myMethod() {
    a && b && c && d && e
}
```

Cyclomatic Complexity Metric Calculation Rules

Start with a initial (default) value of one (1). Add one (1) for each occurrence of each of the following:

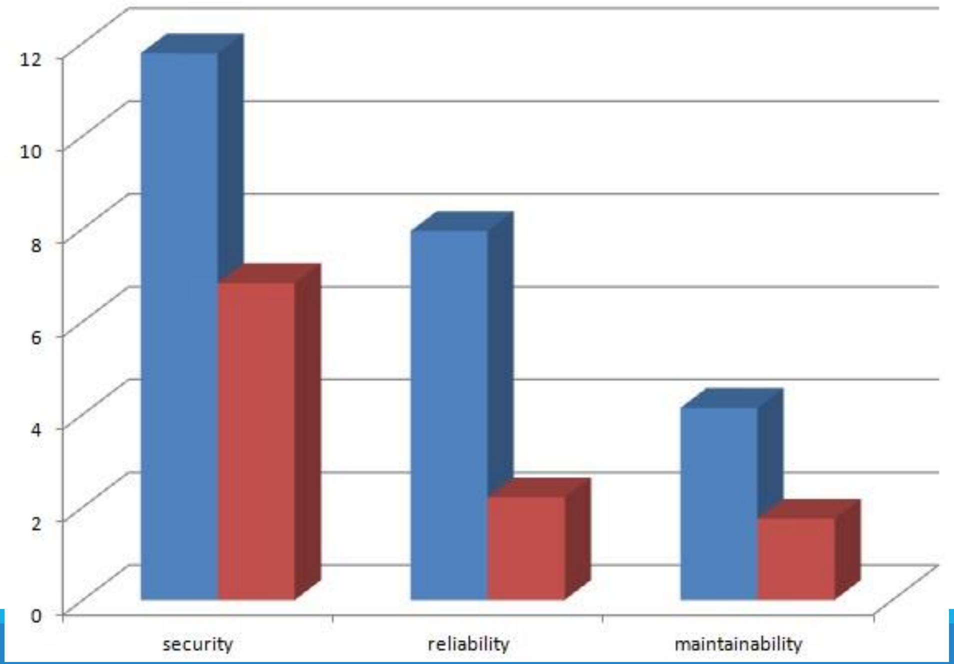
- **if** statement
- **while** statement
- **for** statement
- **case** statement
- **catch** statement
- **&&** and **||** boolean operations
- **?:** ternary operator and **?:** *Elvis* operator.
- **?.** null-check operator

http://codenarc.sourceforge.net

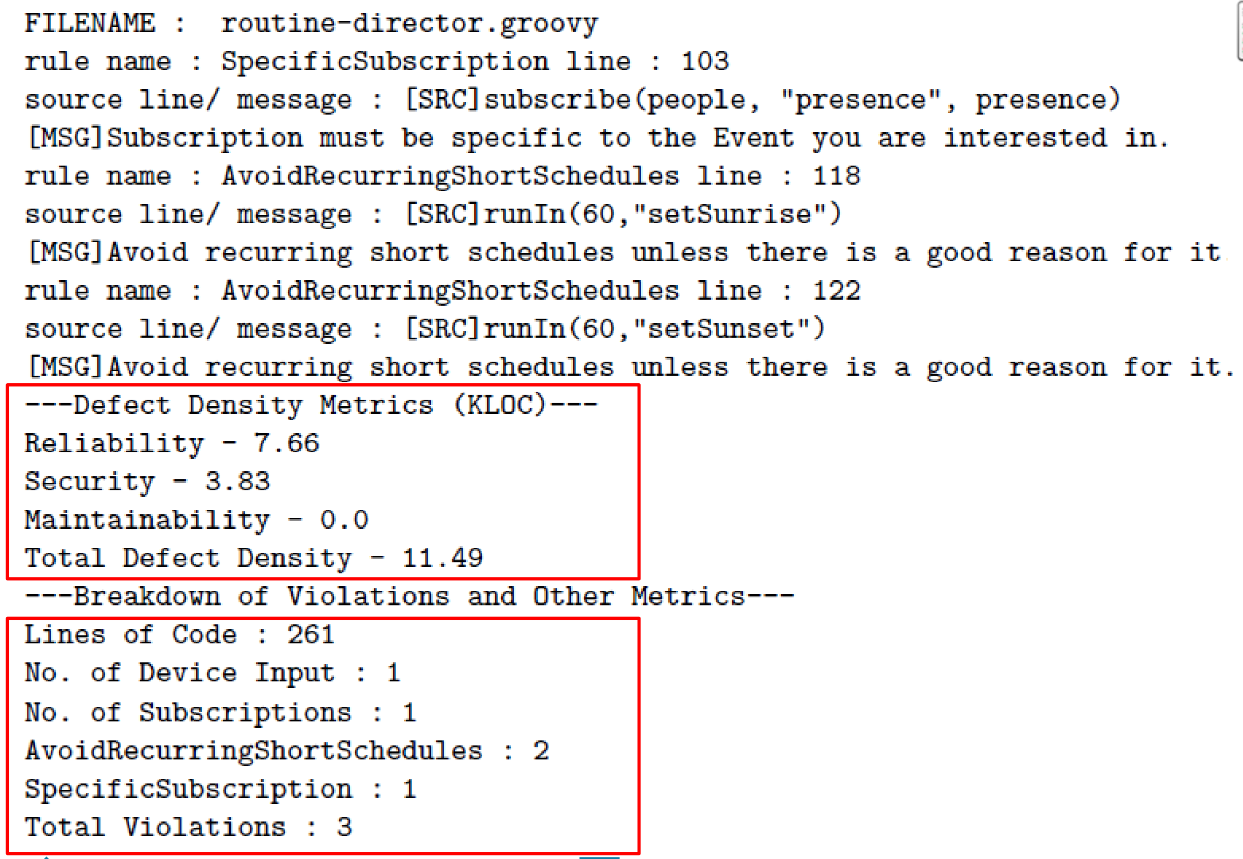
Code Defect Density

- **RQ2**: How do community-created SmartApps differ from official SmartApps in terms of quality?

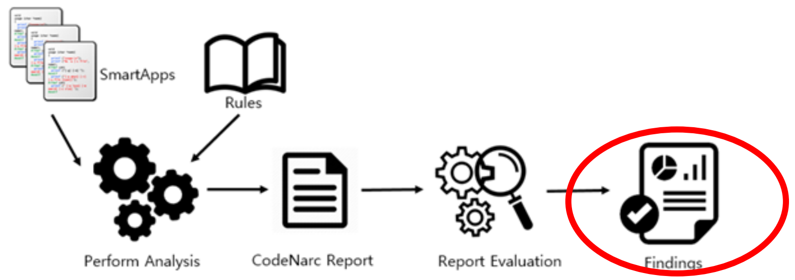
	SmartApp Type	Security	Reliability	Maintainability
🟥	Official	6.83	2.21	1.75
🟦	Community-created	11.78	7.95	4.14



Evaluation Report



```
FILENAME :  routine-director.groovy
rule name : SpecificSubscription line : 103
source line/ message : [SRC]subscribe(people, "presence", presence)
[MSG]Subscription must be specific to the Event you are interested in.
rule name : AvoidRecurringShortSchedules line : 118
source line/ message : [SRC]runIn(60,"setSunrise")
[MSG]Avoid recurring short schedules unless there is a good reason for it
rule name : AvoidRecurringShortSchedules line : 122
source line/ message : [SRC]runIn(60,"setSunset")
[MSG]Avoid recurring short schedules unless there is a good reason for it.
---Defect Density Metrics (KLOC)---
Reliability - 7.66
Security - 3.83
Maintainability - 0.0
Total Defect Density - 11.49
---Breakdown of Violations and Other Metrics---
Lines of Code : 261
No. of Device Input : 1
No. of Subscriptions : 1
AvoidRecurringShortSchedules : 2
SpecificSubscription : 1
Total Violations : 3
```



Factors that affect high defect density:

- Low LOC count – lesser code, lesser error (no error is ideal)
- Same type of error repeated in another line

↑ Code defect density ↓ Quality of code

Official App Code Defect Density

SmartApp	Reliability	Maintainability	Security	Total
presence-change-push.groovy	0	0	71.43	71.43
let-there-be-light.groovy	0	0	62.50	62.50
sleepy-time.groovy	17.86	0	35.71	53.57
turn-it-on-when-im-here.groovy	0	0	47.62	47.62
presence-change-text.groovy	0	0	46.51	46.51
wattvision-manager.groovy	25.64	9.62	9.62	44.87
energy-alerts.groovy	27.4	13.7	0	41.1
turn-on-only-if-i-arrive-aftersunset.groovy	0	0	40	40
light-follows-me.groovy	0	0	38.46	38.46
yoics-connect.groovy	12.17	9.73	14.6	36.5

Typical SmartApp size: 200 lines

Community-created App Code Defect Density

SmartApp	Reliability	Maintainability	Security	Total
smart-energy-service.groovy	99.43	17.21	5.74	122.37
spruce-scheduler.groovy	66.39	18.23	1.87	86.49
initial-state-eventstreamer.groovy	6.58	6.58	69.08	82.24
lights-off-with-no-motion-andpresence. groovy	0	15.15	60.61	75.76
let-there-be-dark.groovy	0	0	58.82	58.82
smart-alarm.groovy	54.14	1.34	0.67	56.82
tcp-bulbs-connect.groovy	47.72	4.34	2.17	54.23
turn-off-with-motion.groovy	0	17.24	34.48	51.72
gideon.groovy	0	40.2	10.05	50.25
obything-music-connect.groovy	50	0	0	50

Research Questions

- **RQ1**: What are the common violations found in SmartApps?
  - ➢ security violations - unspecific subscriptions, web services-related flags; threat to system
  - ➢ convention and size related violations - inverted if-else, cyclomatic complexity; indicates poor maintainability
- **RQ2**: How do community-created SmartApps differ from official SmartApps in terms of quality?
  - ➢ they have higher defect densities in security, reliability, and maintainability compared to official apps; indicates low quality
  - ➢ most community-created apps contain maintainability defects which means that they need to follow conventions and guidelines to produce better software

Summary and Future Work

- Contribution: developed the first automatic code review tool for the quality evaluation of SmartThings Applications
- Analyzed 105 official and 74 community-created apps
- Used an existing static analysis tool, CodeNarc
- Added custom rules for SmartApps
- Used code defect density to measure SmartApp quality

Findings

- Common violations:
  - security violations - unspecific subscriptions, web services-related flags
  - convention and size related violations - indicate poor maintainability
- Both types of SmartApps need improvement – security *
- Community-created apps need to follow the standard Groovy conventions and SmartThings best practices expressed in the guidelines

- Future Work:
  - perform an in-depth analysis on how to evaluate the quality of SmartApps with external services

Thank You