Home, SafeHome: Smart Home Reliability with Visibility and Atomicity

Shegufta B. Ahsan, Rui Yang, Shadi A. Noghabi, and Indranil Gupta

Department of Computer Science, University of Illinois at Urbana-Champaign
Microsoft Research
Smart Home World

*Smart Device:* 1) connected to other devices via wireless protocols  
2) controlled by home automation systems
**Smart Home World**

*Smart Device*: 1) connected to other devices via wireless protocols  
2) controlled by home automation systems

“Humans need to control their lives, not control devices.”

-- Davidoff et al, UbiComp’06
How people control smart home?

- by *Command*
  
e.g. {Make an espresso}

- by *Routine*: a sequence of commands
  
e.g. Prep. Breakfast = {Make an espresso; make a pancake}

*Current systems execute Best-Effort!*
Two Natural Expectations from Users

- Execute everything in a routine – **Atomicity**
  - All commands in the routine need to finish successfully, or none do

- When conflicts happen, people hope routines to execute one after another
  – **Isolation / Serial Equivalence**

*Poorly supported in current systems!*

*Routines are common to be long running, e.g. trash-out routine.*
SafeHome

- Home Automation System that can
  - Support *long running* routines
  - Properly *isolate* concurrent routines (providing *serial equivalence*)
  - Ensure routine execution *atomicity*

- Key challenge: Actions are visible to users

- Methodology:
  - Four *Visibility Models* (Spectrum for user choices)
  - *Lock-based* mechanism with *leasing* design
Visibility Models

Four Visibility Models:
- Weak, Eventual, Partitioned Strict, Global Strict

Example Scenarios: 5 routines are initiated *simultaneously* on 4 devices

3 Routines Initiated by User:
- Coffee Maker
- Pancake maker

R1: (espresso)
R2: (americano)
R3: (plain)

2 Routines triggered by other sensors:
- Vacuum
- Mopper

R4: (living room)
R5: (kitchen)
Weak Visibility (WV) Model -- Status Quo

Strategy:
- Execute routine immediately when triggered

Insertion time

Parallel Execution

Two commands send simultaneously to one device may cause errors.

Finish in 2 time units

Coffee maker
Pancake maker
Vacuum
Mopper
Global Strict Visibility (GSV) Model

Strategy:
- Execute at most one routine at a time

Insertion time

- Strongest Visibility Model
- Example Usage: resource constrained environment:
  - e.g. 1000-watt max supply < coffee maker 600W + pancake maker: 600W
Partitioned Strict Visibility (PSV) Model

Strategy:
- Routines touching disjoint devices do not block each other
- Useful when routines need to execute without interference through duration.
- Might still takes long with long running routines.
Strategy:
- Routines can concurrently execute *without violating some serial order*.
Eventual Visibility (EV) Model

Strategy:
- Routines can concurrently execute *without violating some serial order*.
- Each routine holds the **locks** for devices it touches (but can *lease the lock*).

```
Managed at central device (e.g. hub)
```

**No code needed**

- **Pre-lease**
- **Post-lease**
Eventual Visibility (EV) - Post-Lease

Post-lease:
- If a routine is done with a device $D$, it can post-lease $D$'s lock to another routine.

Serial order:
lessor $\rightarrow$ lessee
($R1 \rightarrow R2$)

Insertion time is followed by time.

$R1$ will be done with coffee maker.
Eventual Visibility (EV) - Pre-Lease

Pre-lease:
- If a routine has acquired the lock but not accessed a device $D$, it can pre-lease $D$'s lock to another routine.

**Insertion time**

**R1**

**R2**

**R3**

**time**

**pre-lease**

$R1$ will start to access pancake maker

Serial order: lessee $\rightarrow$ lessor

( $R3$ $\rightarrow$ $R1$ )
Eventual Visibility (EV)

EV finishes routine
- with short wait and provides serial equivalence
- with higher temporary incongruence: intermediate state is not serially equivalent

Finish in 3 time units

pancake and coffee maker can not be both ON under any serial order
Eventual Visibility (EV) - Lineage Table

**Lineage Table**: SafeHome's plan of which routine will access which device.

- **R1[A]**: Get lock Access
- **R2[S]**: Routine Scheduled
- **R3[A]**: Lock Leased out
- **R4[A]**: Lock Released
- **R5[R]**: Routine is triggered

**Scheduling plan placement**:  
- Placed when routine is triggered  
- Use *backtracking* for valid placement  
- Explore two other policies (FCFS, JiT)
Failure Serialization and Rollback

Device might *fail*:
- *Rollback*? Try to *serialize* the failure/restart event!
- If the failed device is not touched by the routine:
  - *Arbitrary* Serial Equivalence order
- If device fails/restarts after the last touch:
  - *Routine* → *Fail/Restart* Serial Equivalence order
- If device fails/restarts before the first touch:
  - *Fail/Restart* → *Routine* Serial Equivalence order
- If device fails/restarts during the touch:
  - *Rollback* routine
SafeHome Implementation

Implementation
- ~2k line of Java code
- Support long running routine expression (JSON)
- Popular Smart Device integration (TP-link, Wemo)

Experiment Setup
- Deployment & Simulation
- Real-world Benchmark
  - Derived from IoTBench Test Suite
  - Morning, Party, Factory Scenario
- Workload-Driven
  - Average of 500k runs
Real-World Benchmark

Temporary Incongruence: the ratio of time when intermediate state is not serially equivalent.

Final Incongruence: the ratio of runs that end up in an incongruent state.
Pre/Post leases reduce the E2E latency (user-facing metrics) with the cost of Temporary Incongruence.
Takeaways

- Safehome is a first step to provide reliability from routine execution level

- SafeHome provides four Visibility Models (WV, EV, PSV, and GSV)

- Eventual Visibility (EV) model provides the best of both worlds, with:
  - Good user-facing responsiveness (0 - 23.1%)
  - Strongest end state congruence equivalent guarantee (as GSV)

- Lock-leasing improves latency by 1.5X - 4X

For questions: contact author Rui Yang <ry2@illinois.edu>