Flask

- Problem: writing stream processing programs for sensor networks (TinyOS / NesC) is hard (to debug, maintain, etc)
- Solution: provide a high-level language for developers that abstracts away the complexity of communications and parallelism

Flask Programming Model

- Each “operator” performs operations on one or more “streams” using “stream combinators”
- Developers can define task graphs for each operator programmatically, using an Ocaml-like language
- If necessary, it is possible to define code blocks in NesC for structures and processes

Communications

- IPC: Developer “wires” the streams and combinators together, and the individual tasks are fired by the posting “wire”
- Network: uses a communications framework called “flows”, based on pub / sub
- Developer simply ties operators together with flow Ids, Flask takes care of the rest

Evaluation

- “Ease of use”... produces shorter code...
- Implemented seismic event detection and TinyDB using Flask
- Overhead: Flows imposes a relatively large overhead on comms, and also on memory
- Results from running a query seem a little erratic – however probably not attributable to Flask

Comments / Questions