

Enabling Development Practices in a Remote Location

Experiences from the ODC
Project

We are so used to co-located communication that we often don't recognise that distributed communication is a distinct skill



Traditional learning methods break down in distributed teams

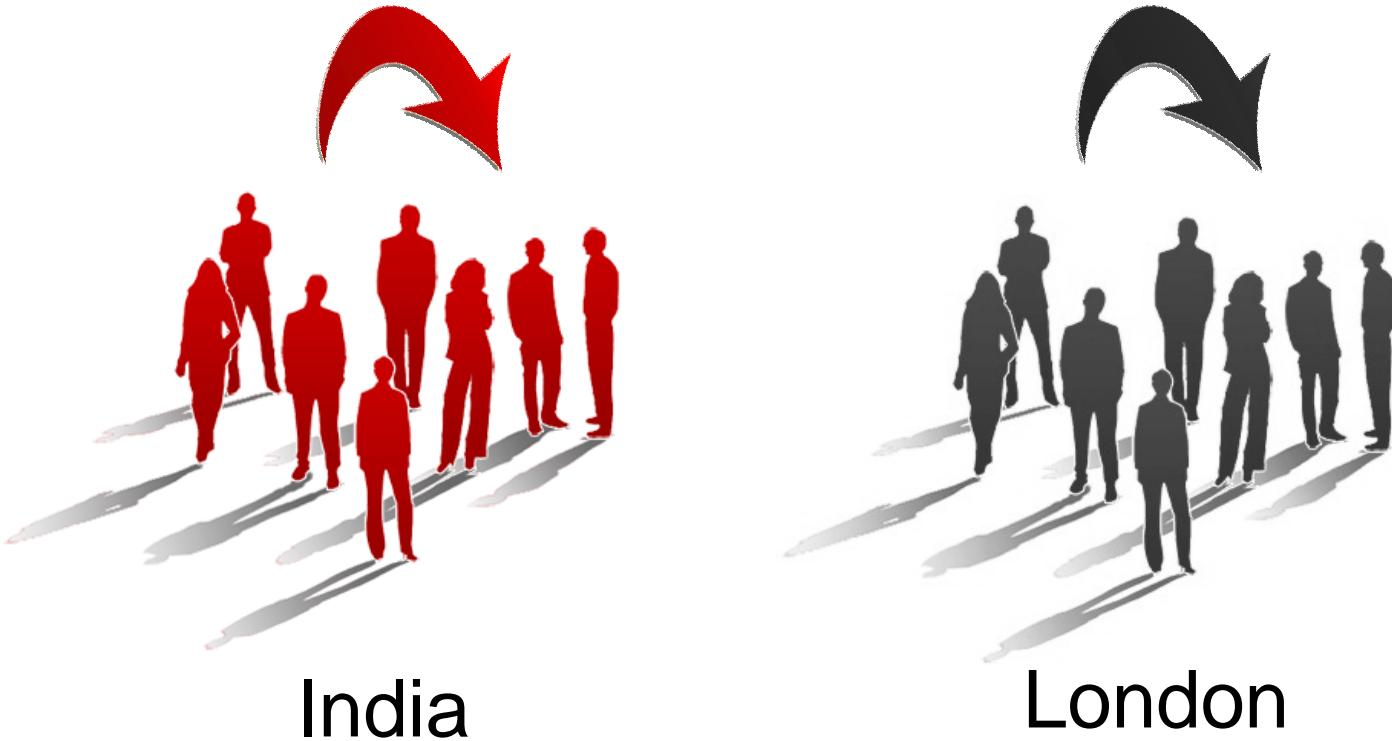


Successful practices foster the idea of a single distributed team



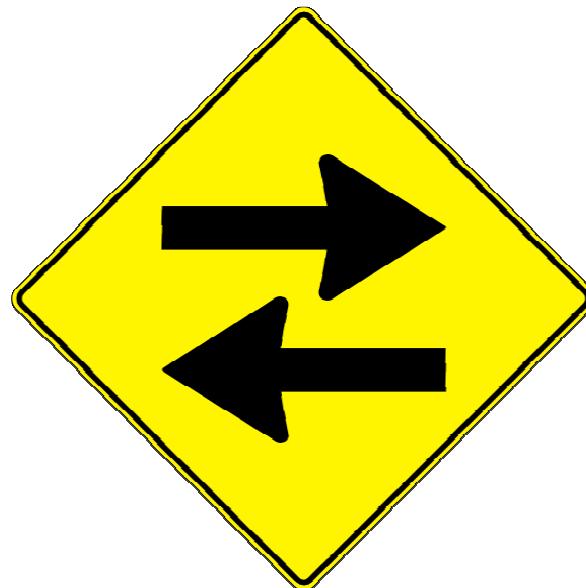
The Agile Backlog

One Story List



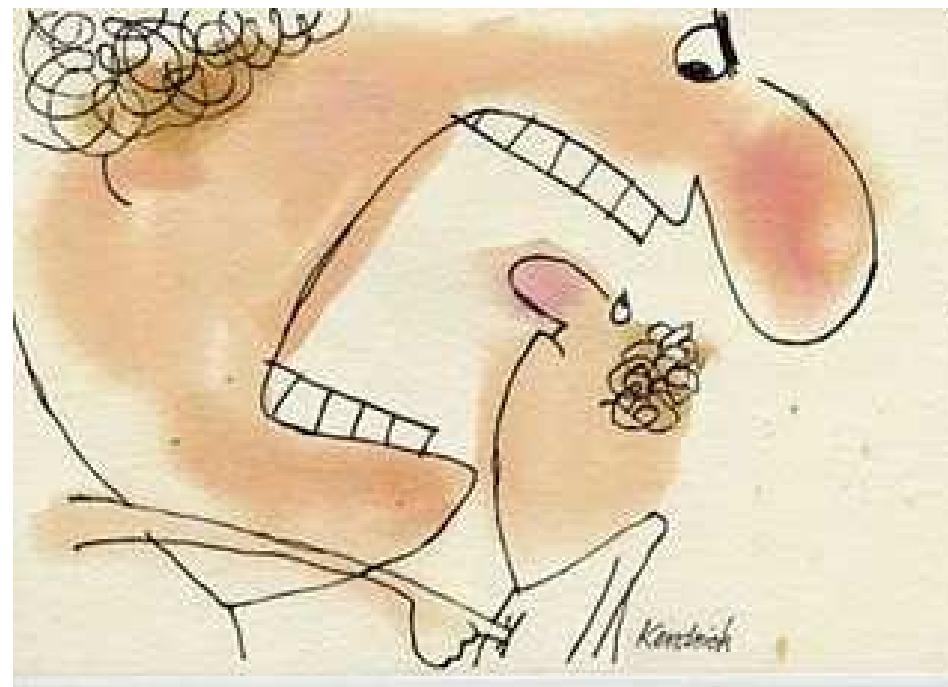
Key Tenet (1)

Learning practices should be collaborative and bi-directional



Learning practices should be collaborative and bi-directional

One way ‘instruction’ can have a negative affect on morale



Learning practices should be collaborative and bi-directional

In corporate contexts collaborating to solve a problem works better than teaching, as it is empowering



Learning practices should be collaborative and bi-directional

Group forums are more appropriate for feedback as the potential for offence is diluted by the group



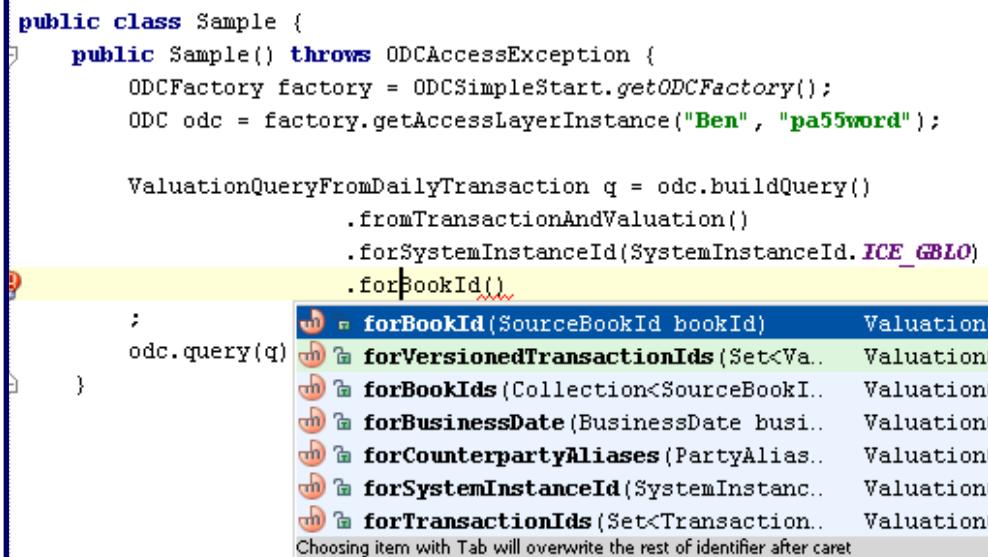
Key Tenet (2)

The code base should be the main tool for communicating practices.

```
public class Sample {
    public Sample() throws ODCAccessException {
        ODCFactory factory = ODCSimpleStart.getODCFactory();
        ODC odc = factory.getAccessLayerInstance("Ben", "pa55word");

        ValuationQueryFromDailyTransaction q = odc.buildQuery()
            .fromTransactionAndValuation()
            .forSystemInstanceId(SystemInstanceId.ICE_GBL0)
            .forBookId();
    }

    odc.query(q);
}
```



The screenshot shows a Java code editor with the following code:

```
public class Sample {
    public Sample() throws ODCAccessException {
        ODCFactory factory = ODCSimpleStart.getODCFactory();
        ODC odc = factory.getAccessLayerInstance("Ben", "pa55word");

        ValuationQueryFromDailyTransaction q = odc.buildQuery()
            .fromTransactionAndValuation()
            .forSystemInstanceId(SystemInstanceId.ICE_GBL0)
            .forBookId();
    }

    odc.query(q);
}
```

A tooltip is displayed over the line ".forBookId();", listing several method suggestions:

- forBookId(SourceBookId bookId) Valuation
- forVersionedTransactionIds(Set<Va.. Valuation
- forBookIds(Collection<SourceBookI.. Valuation
- forBusinessDate(BusinessDate busi.. Valuation
- forCounterpartyAliases(PartyAlias.. Valuation
- forSystemInstanceId(SystemInstanc.. Valuation
- forTransactionIds(Set<Transactio.. Valuation

At the bottom of the tooltip, the text "Choosing item with Tab will overwrite the rest of identifier after caret" is visible.

The code base should be the main tool for communicating practices.

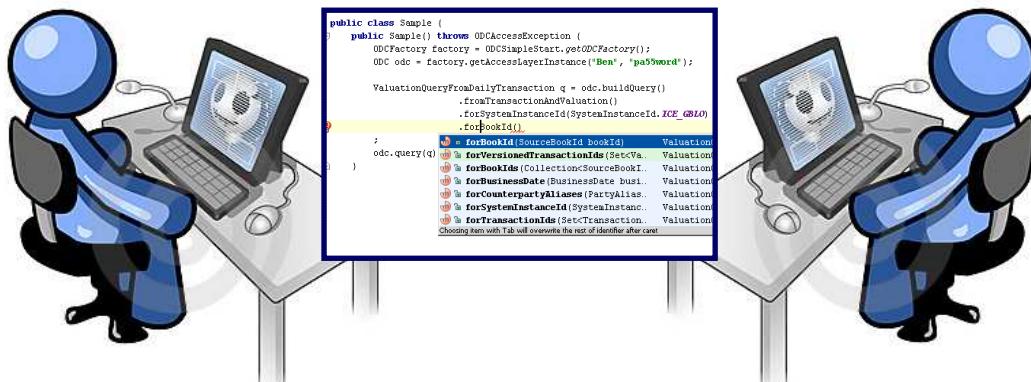
Code is language and culturally neutral



The code base should be the main tool for communicating practices.

The codebase facilitates contextual learning

Teaching patterns and practices is significantly more successful if it is done in the context of real work in a real code base.



```
public class Sample {  
    public Sample() throws ODCAccessException {  
        ODCFactory factory = ODCSimpleStart.getODCFactory();  
        ODC odc = factory.getAccessLayerInstance("Ben", "pa5word");  
  
        ValuationQueryFromDailyTransaction q = odc.buildQuery()  
            .fromTransactionAndValuation()  
            .forSystemInstanceId(SystemInstanceId.ICE_GLO)  
            .toBookId();  
  
        ;  
        odc.query(q);  
    }  
}
```

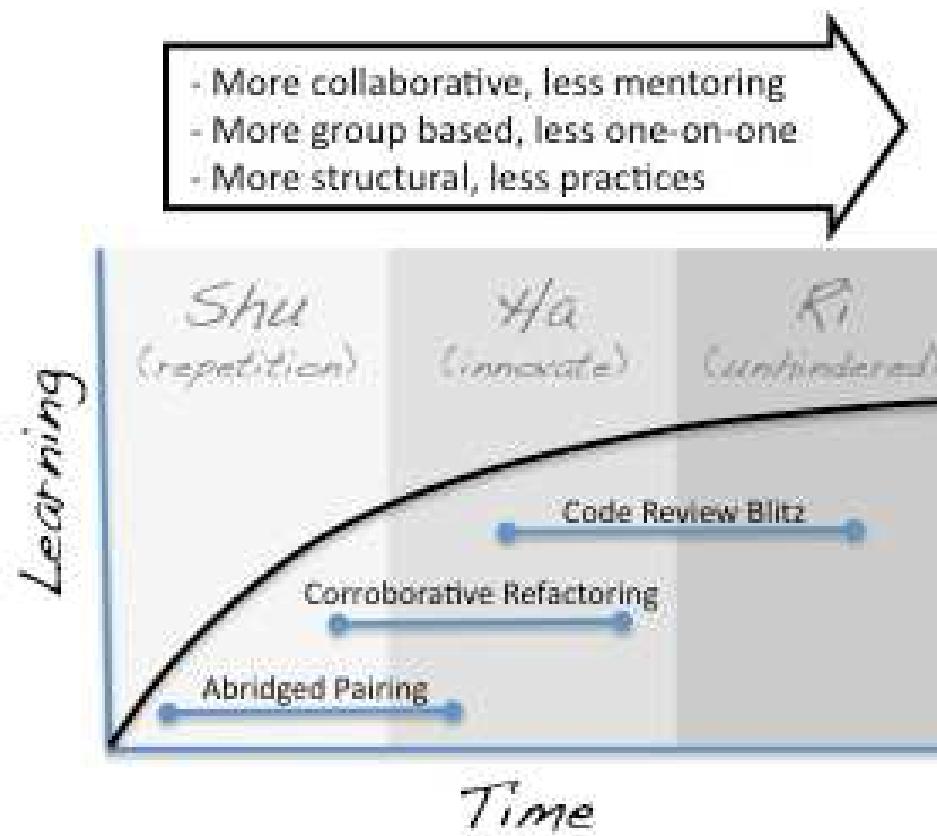
Key Tenet (3)

The practices should change as the team evolves

You need different ‘tools’ at different times

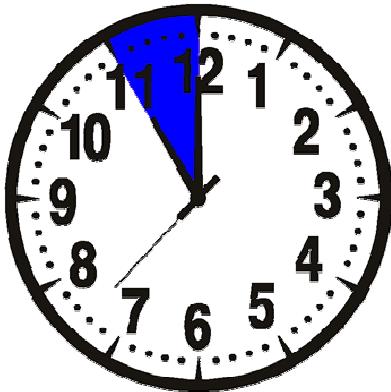


High intensity, one-on-one practices work well at the start

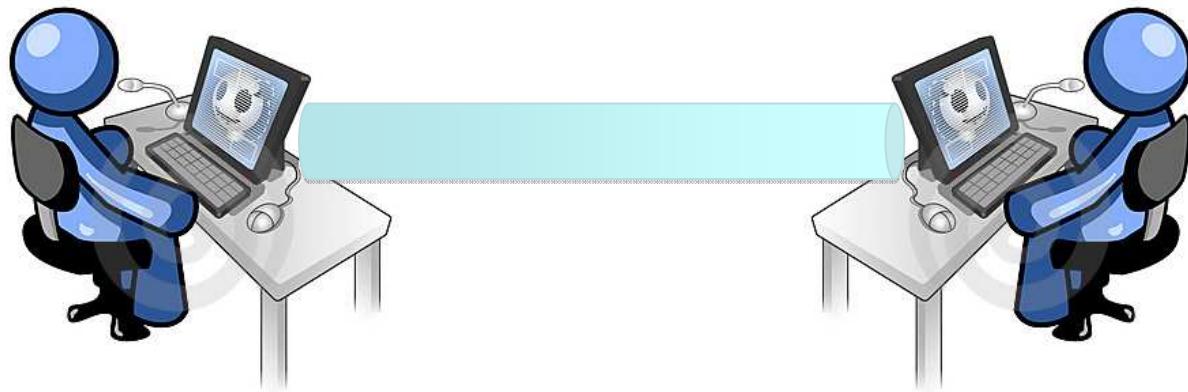


More collaborative, group based practices work well as the team matures

Abridged Pairing



One hour per day



Pros:

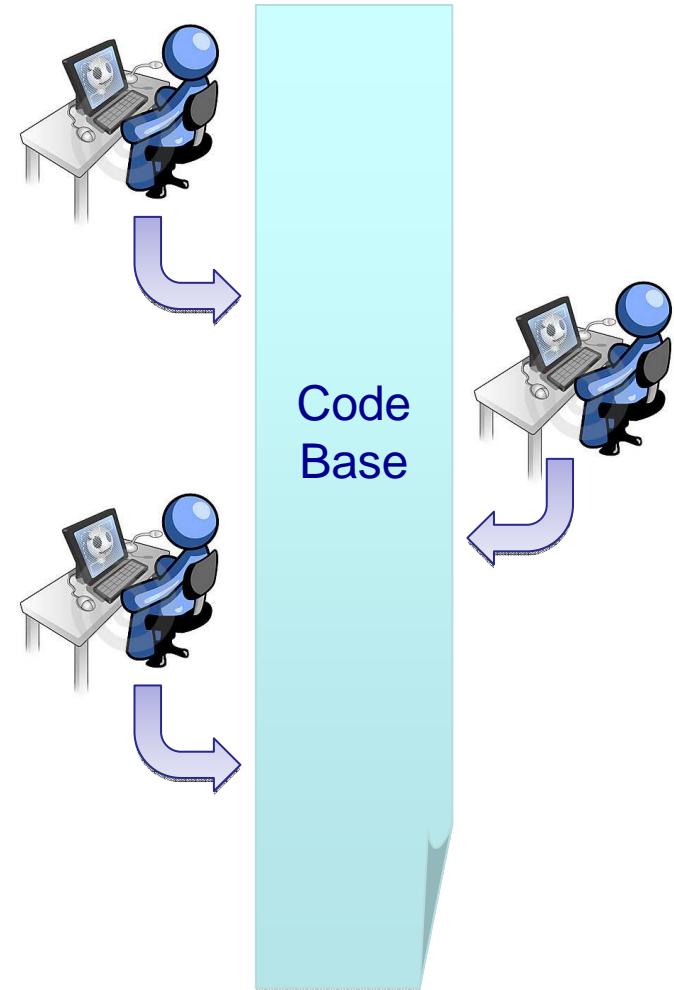
- Very successful mentoring practice
- Targeted and feedback driven
- Instruction is provided in the context of a real work problem
- ⇒One of the best ways to teach skills like OO and test driven development

Cons:

- Unidirectional
- Can be frustrating for both parties
- Both parties can't edit code concurrently (software limitation) meaning it's not really a communication conduit.
- ⇒Best for a short period of enablement

Collaborative Refactoring

- Offline practice
- Similar to a traditional code review but is done during the development of a story.
- Reviewer actually refactors sections and then talks them through with the original developer. The original developer does the same.



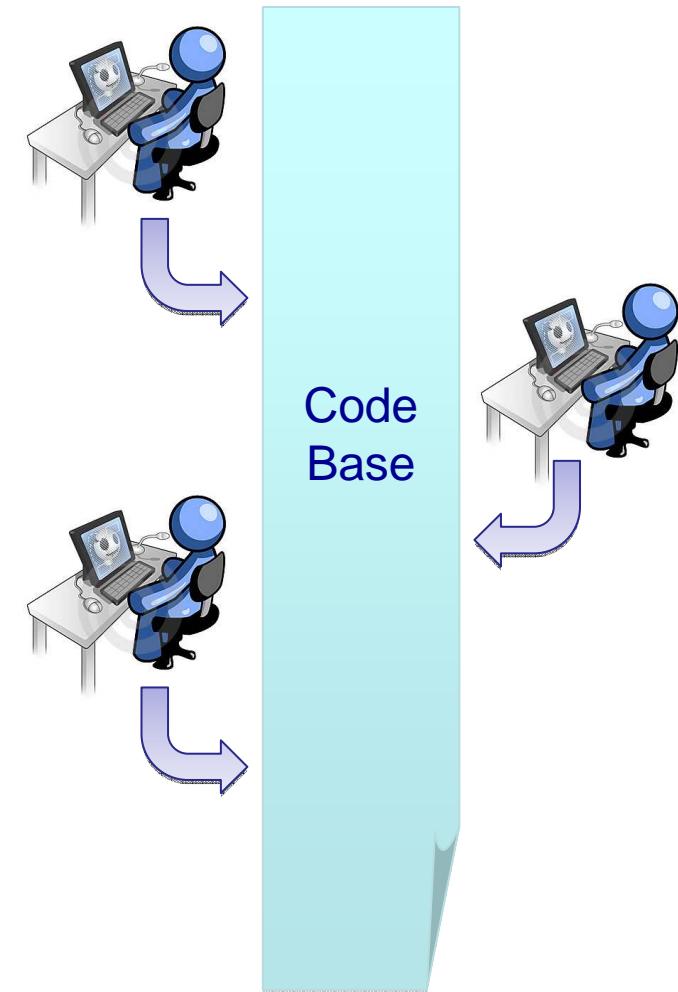
Collaborative Refactoring

Pros:

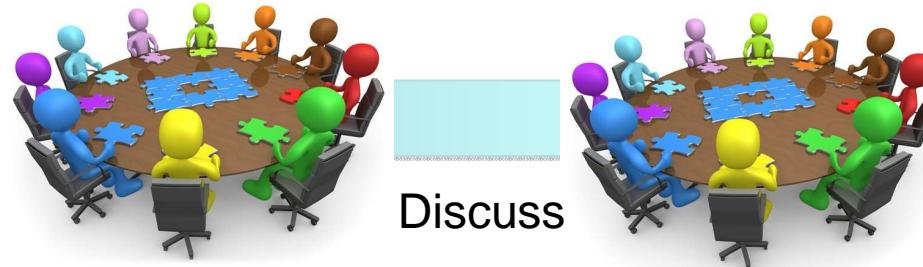
- Real world problems
- More interactive for the reviewer
- The code is the primary communication channel

Cons:

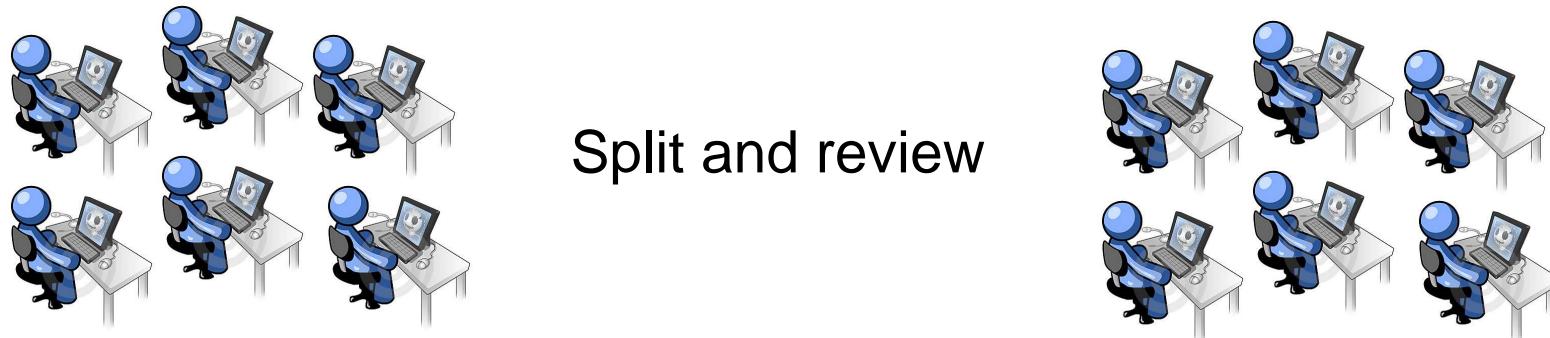
- Time consuming
- Unintentional offense



Code Review Blitz



Discuss



Split and review



Discuss

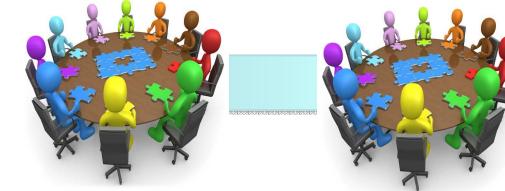
Code Review Blitz

Pros:

- Group provides momentum
- Groups are a better forum for feedback
- Collect boarder themes for further discussion or follow up

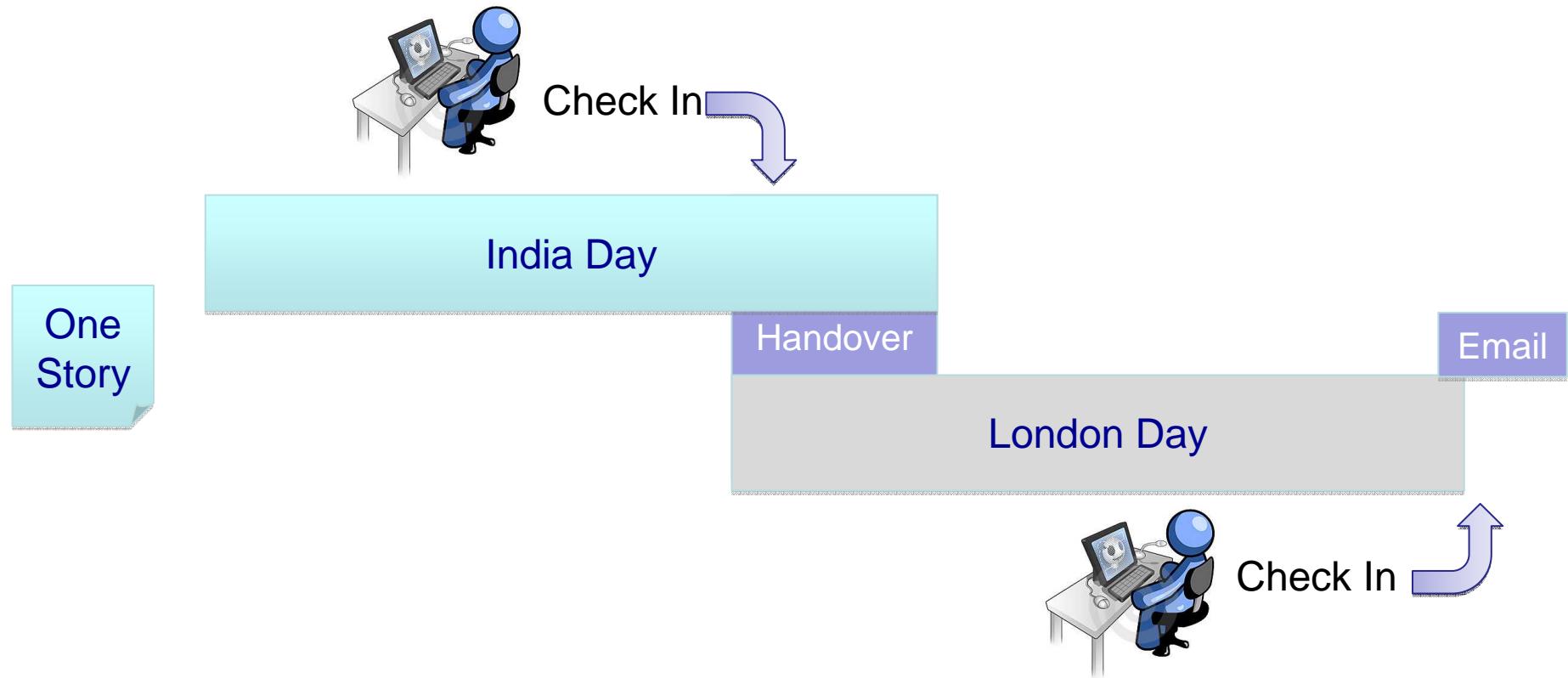
Cons:

- Lack of Freshness: The code being reviewed can be out of date
- More code review than collaborative improvement



Follow-the-Sun Pairing

(work in progress)



Developer rotations: the *best* way to teach practices



Practice Champions

Inculcate a remote team member with a certain skill for them to distribute



Building Rapport

Video Conferences without specific agendas



Summary

Bidirectional,
collaborative instead
of purely instructional

Try to use the code
base to communicate
instead of just the
phone

Different practices
are needed at
different times, no
one will do