KEEPING YOUR DATABASE AND PHP IN SYNC

MAGGIE NELSON
MAY 22, 2008
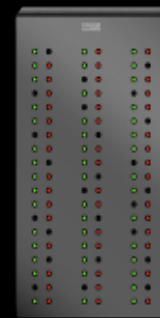
schematic

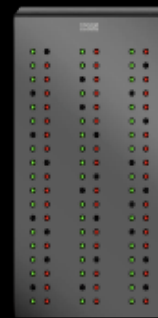dev   prod
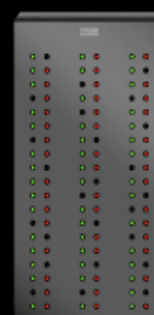
dev → prod

dev → qa → prod

dev → qa → staging → prod

dev1
(new stuff)

dev2
(new stuff)

dev3
(maintenance)

qa1
(new stuff)

qa2
(maintenance)
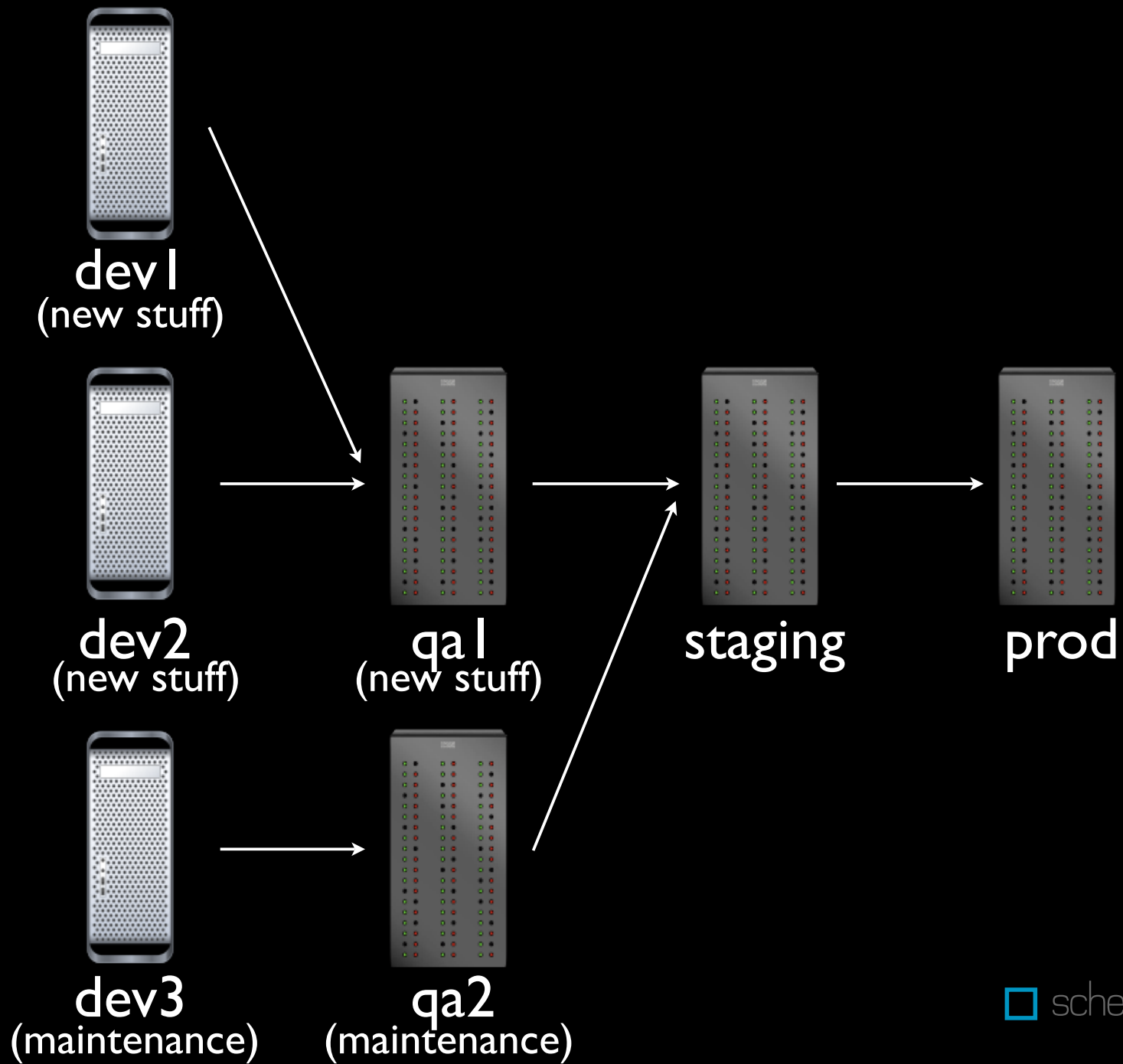
staging

prod

schematic

PHP and DB are separate

= PHP + database

= PHP + database

schematic

releaseNumber.sql

```sql
-- release.2.0.sql

create table friends (
    user_id not null
        constraint friend_user_fk
        references users on delete cascade,
    friend_user_id not null
        constraint friend_friend_user_fk
        references users on delete cascade,
    created date default sysdate not null
);
```

schematic

release 2.0

PHP + database

cut 2.0 release at svn revision 207 → run release.2.0.sql in database → install 2.0 code

schematic

reverting release 2.0

PHP

+
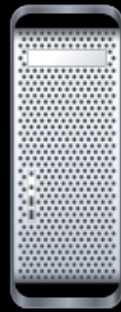
database

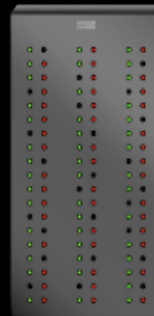revert to previous version of code (prior to svn revision 207) → ??? in database

schematic
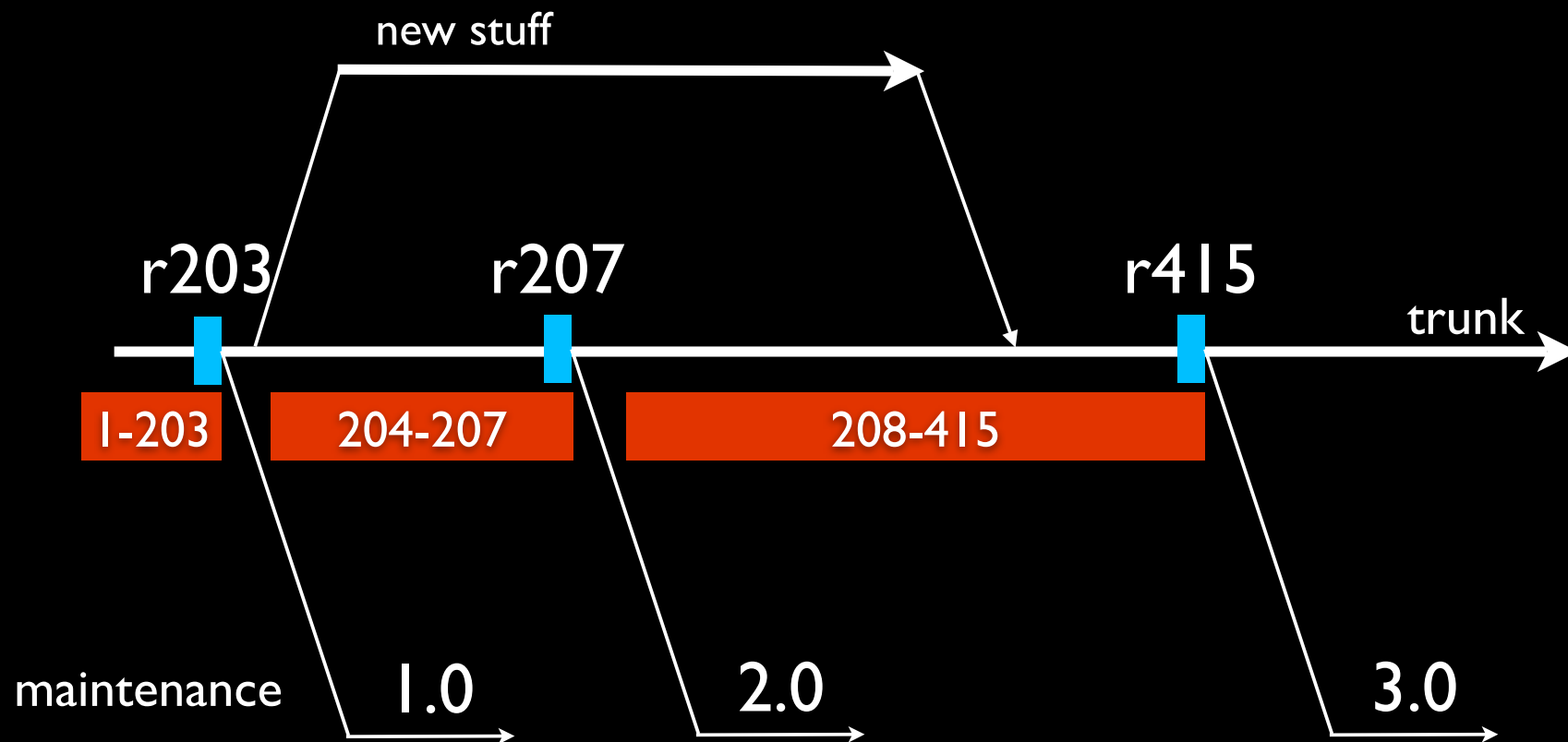
releaseNumber.undo.sql

```sql
-- release.2.0.sql

create table friends (
    user_id not null
        constraint friend_user_fk
        references users on delete cascade,
    friend_user_id not null
        constraint friend_friend_user_fk
        references users on delete cascade,
    created date default sysdate not null
);
```

```sql
-- release.2.0.undo.sql

drop table friends;
```

schematic

reverting release 2.0

PHP

+
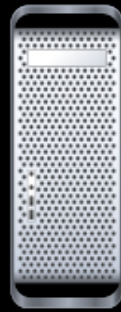
database

revert to previous version of code (prior to svn revision 207) → run release.2.0.undo.sql in database

schematic

Frequent database changes during release cycle

Frequent changes from dev –> QA
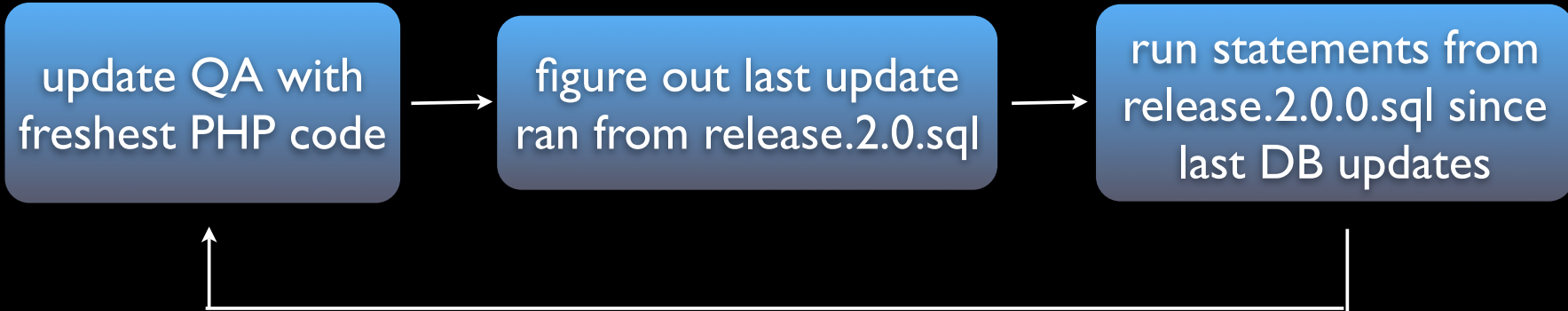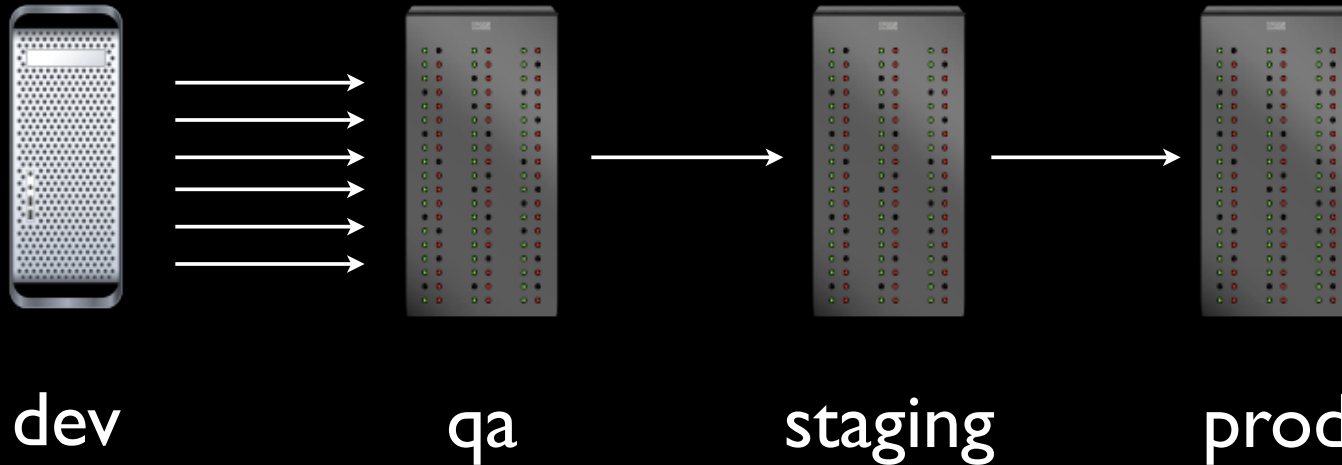
dev   qa   staging   prod

update QA with freshest PHP code	→	figure out last update ran from release.2.0.sql	→	run statements from release.2.0.0.sql since last DB updates

schematic

releaseNumber.sql

```sql
-- release.2.0.sql

-- added Monday
create table friends (...);
create unique index friends_uk
    on friends (user_id, friend_user_id);

-- added Wednesday
create table coupons (...);
create unique index coupons_uk
    on coupons (id);
```

schematic

releaseNumber.sql

```sql
-- release.2.0.sql

update coupon
    set expired = true;

insert into coupon (
    id,
    percent_off)
values (
    'PHP_TEK',
    100);
```

schematic

Communication between developers

dev1
(new stuff)

dev2
(new stuff)

dev3
(maintenance)

qa1
(new stuff)

qa2
(maintenance)

staging

prod

schematic

devs making DB changes

dev1   dev2

| update devX with freshest PHP code | → | figure out last update ran from release.2.0.sql | → | run statements from release.2.0.0.sql since last DB updates |

schematic

```
dev1 writes new code
        |
        v
dev1 commits code  ----->  database changes?  --YES-->  dev1 sends e-mail to dev2 with
                                  |                      instructions for the DB change
                                  NO                              |
                                  |                               v
                                  v                        dev2 checks email?  --NO-->  dev2: svn up
dev2: svn up to get                                               |                          |
new code                                                          YES                        |
        |                                                          |                         |
        v                                                          v                         |
dev2 has new code  <--YES--  dev2 reads instructions              |                          |
                                                                   v                         v
                             dev2 follows instructions correctly?  --NO-->  dev2 environment is broken
                                                                                    |
                                                                                    v
                                                                                EPIC FAIL
```

dev1 writes new code

dev1 commits code

database changes?

YES

dev1 sends e-mail to dev2 with instructions for the DB change

NO

dev2: svn up to get new code

dev2 checks email?

NO

dev2: svn up

YES

dev2 reads instructions

dev2 has new code

YES

dev2 follows instructions correctly?

NO

dev2 environment is broken

EPIC FAIL

releaseNumber.sql

```sql
-- release.2.0.sql

-- added Monday
create table friends (...);
create unique index friends_uk
    on friends (user_id, friend_user_id);

-- added Wednesday
create table coupons (...);
create unique index coupons_uk
    on coupons (id);
```

schematic
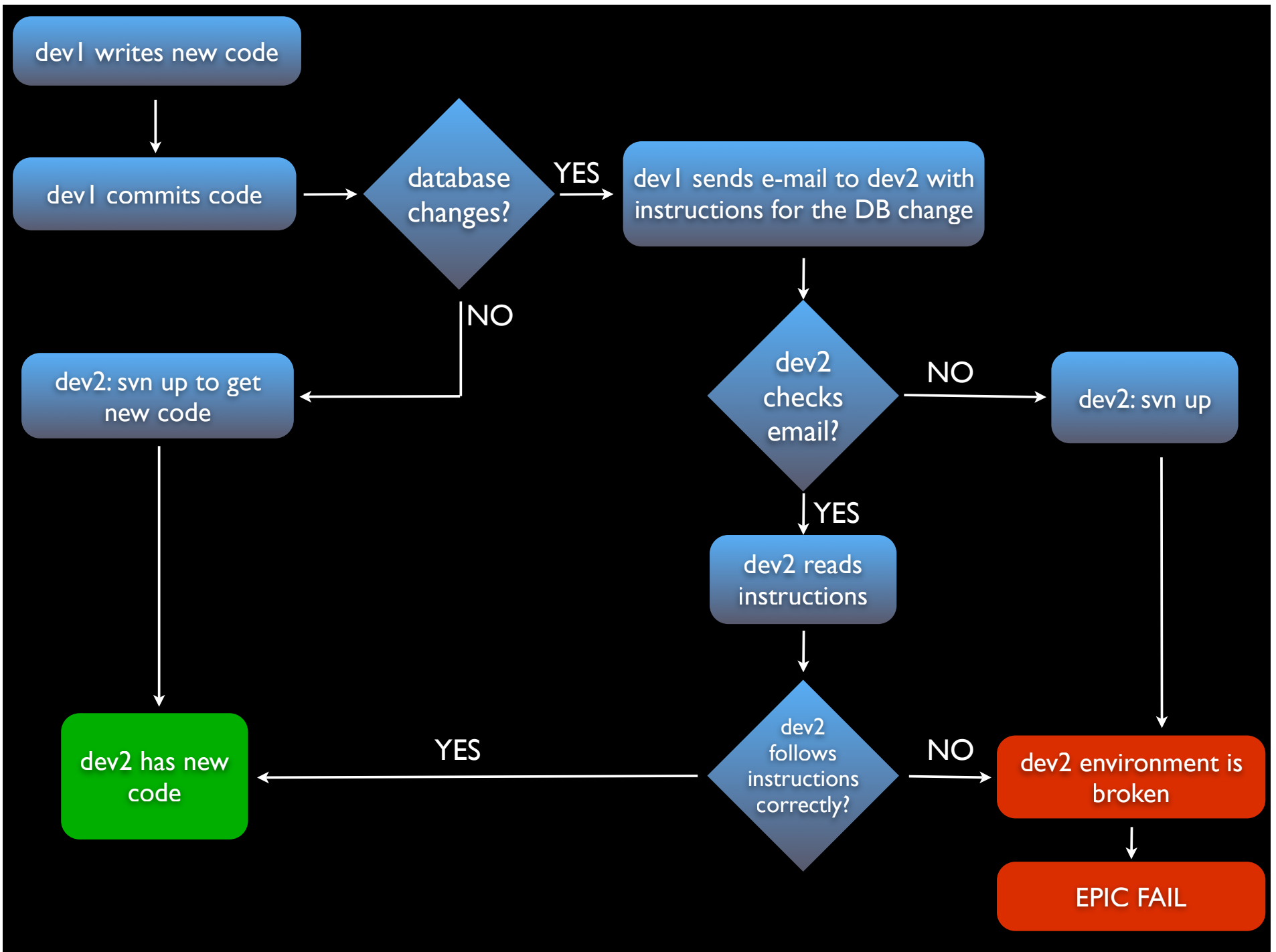
DB deltas

```
-- delta 1

-- begin up:
create table friends (...);
create unique index friends_uk
    on friends (user_id, friend_user_id);
-- end up

-- begin down:
drop index friends_uk;
drop table friends;
-- end down
```

```
-- delta 2

-- begin up:
create table coupons (...);
create unique index coupons_uk
    on coupons (id);
-- end up

-- begin down:
drop index coupons_uk;
drop table coupons;
-- end down
```
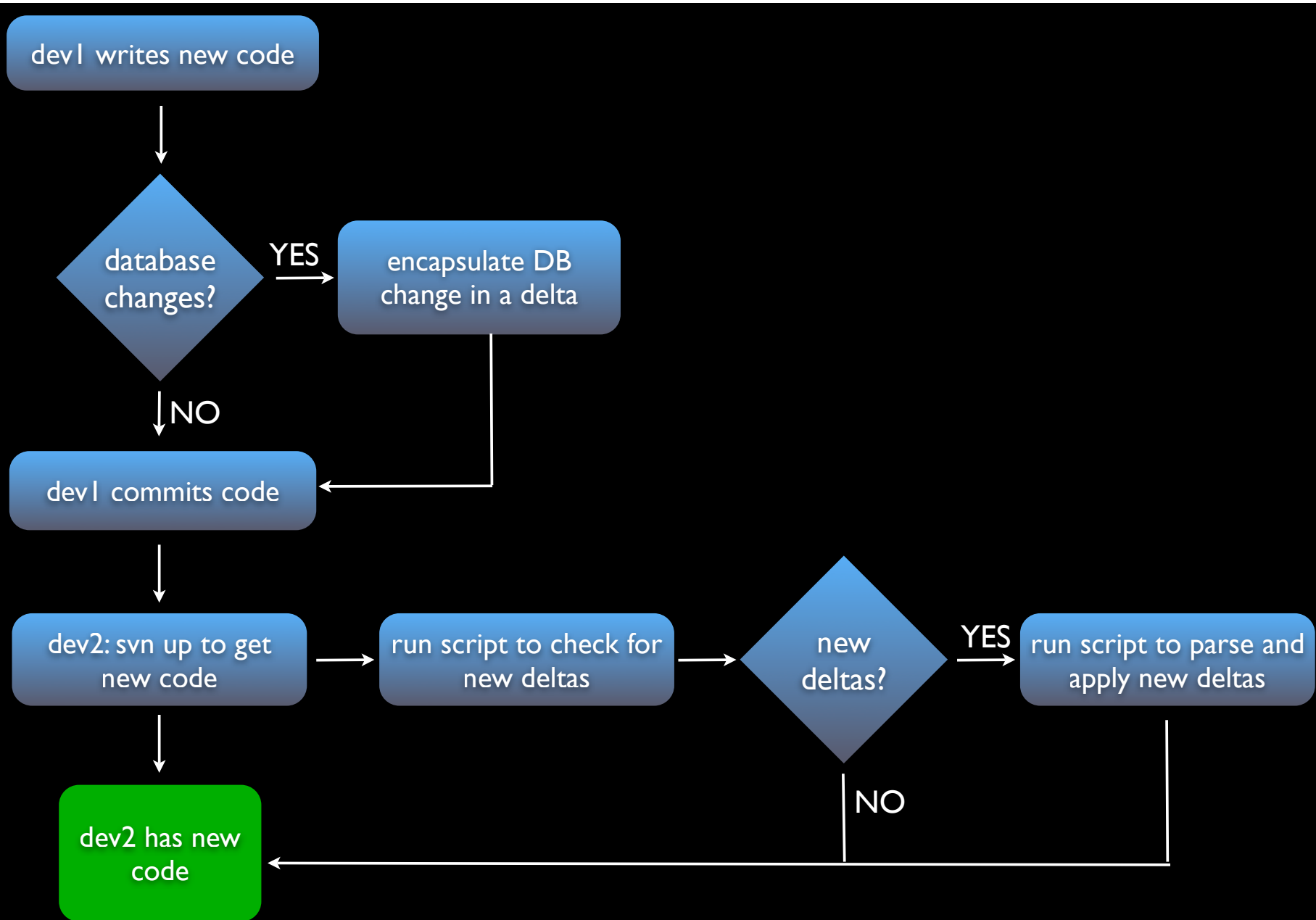
DB deltas as XML

```xml
<?xml version="1.0"?>
<!DOCTYPE delta SYSTEM "delta.dtd">

<delta number="1">
  <summary>friends stuff</summary>

  <up><![CDATA[
    create table friends (...);
    create unique index friends_uk
        on friends (user_id, friend_user_id);
  ]]></up>

  <down><![CDATA[
    drop index friends_uk;
    drop table friends;
  ]]></down>
</delta>
```

Stored procedures

Calling stored procedures from PHP

```php
<?php

    $dbh = new PDO('oci:', 'scott', 'tiger');

    $dbcode =
        "begin
            friend_pkg.add_friend(
                p_user_id => :user_id,
                p_friend_user_id => :friend_user_id
            );
        end;";

    $stmt = $dbh->prepare($dbcode);
    $stmt->bindParam(':user_id', $userId);
    $stmt->bindParam(':friend_user_id', $friendUserId);

    $stmt->execute();
?>
```

schematic

Oracle example – pkg header

```
create or replace package friend_pkg as

procedure add_friend(
    p_user_id users.id%type,
    p_friend_user_id users.id%type
);

end;
```

Oracle example – pkg body

```sql
create or replace package body friend_pkg as

procedure add_friend(
    p_user_id users.id%type,
    p_friend_user_id users.id%type
)
is
begin
    if p_user_id <> p_friend_user_id then
        insert into friends (
            user_id,
            friend_user_id)
        values (
            p_user_id,
            p_friend_user_id);
    end if;

    -- Publish to newsfeed news about a new friend!
    begin newsfeed_pkg.added_friend(p_user_id, p_friend_user_id); end;

    -- Give new friend rights to view private comments!
    begin comments_pkg.view_private(p_friend_user_id, p_user_id); end;

    -- etc.
end;

end;
```
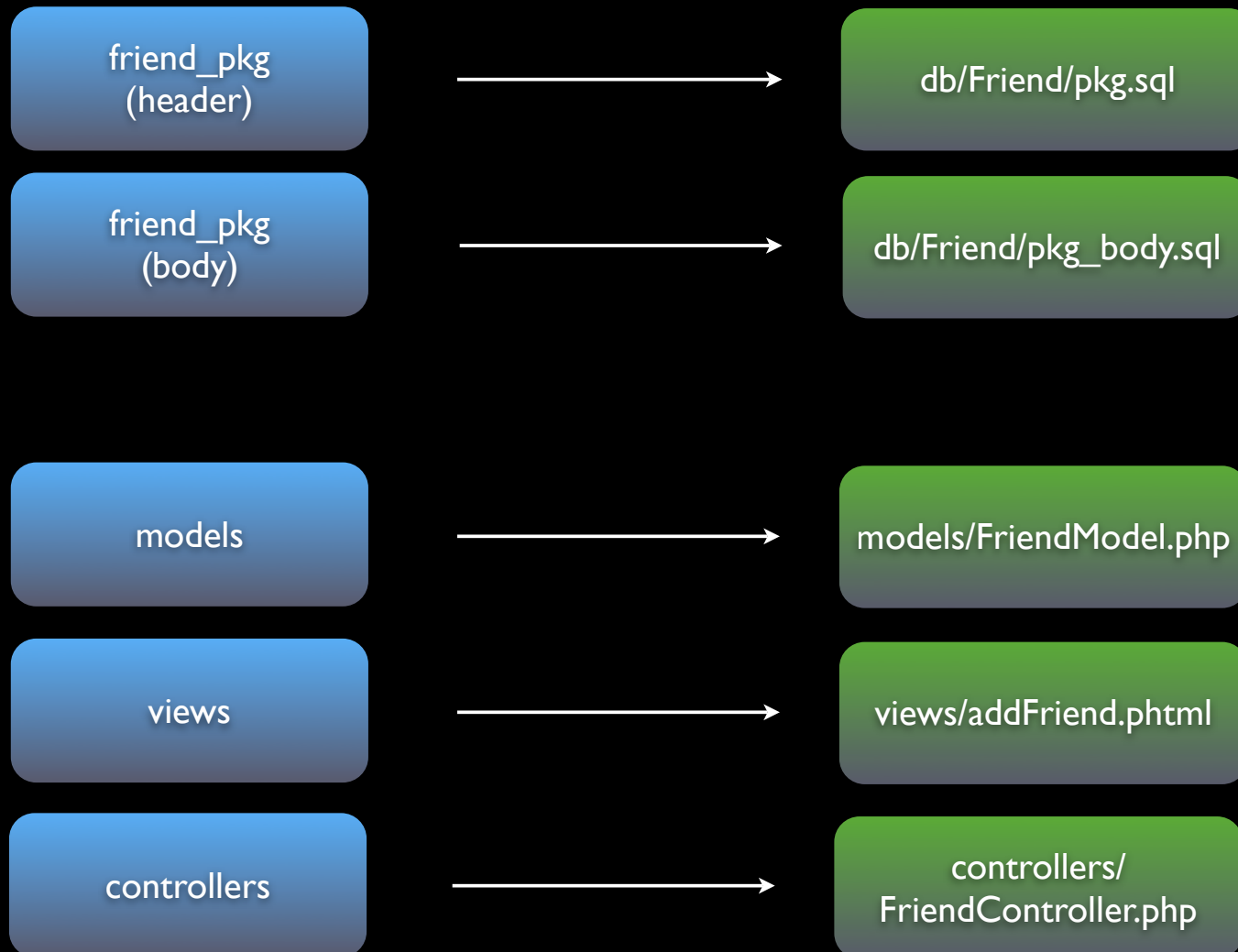
Stored procedures in deltas

```xml
<?xml version="1.0"?>
<!DOCTYPE delta SYSTEM "delta.dtd">

<delta number="1">
  <summary>friends stuff</summary>

  <up><![CDATA[
    -- paste package header
    -- paste package body
  ]]></up>

  <down><![CDATA[
    -- paste package header of previous revision
    -- paste package body of previous revision
  ]]></down>
</delta>
```

schematic

Reference, don't copy

```xml
<?xml version="1.0"?>
<!DOCTYPE delta SYSTEM "delta.dtd">

<delta number="1">
  <summary>friends stuff</summary>
  <up><![CDATA[-- some DDL + DML]]></up>
  <down><![CDATA[-- some DDL + DML]]></down>

  <packages>
    <package>
      <header>Friend/pkg.sql</header>
      <body>Friend/pkg_body.sql</body>
    </package>
  </packages>
</delta>
```

schematic

svn cat -r[some_revision] Friend/pkg.sql >> runThis.sql

svn cat -r[some_revision] Friend/pkg_body.sql >> runThis.sql

[some_revision] ??

encapsulate DB changes (DDL + DML) in a delta
(e.g. delta7.xml)

make stored procedure changes

reference stored procedure changes in the delta

make PHP changes

commit PHP changes, stored procedure
changes and the new delta7.xml

svn log -r PREV:HEAD delta7.sql

----
r107 | maggie_n | 2008-02-25 15:13:50 -0500 (Mon, 25 Feb 2008) | 15 lines

svn log -r PREV:HEAD delta7.sql --xml

```xml
<?xml version="1.0"?>
<log>
<logentry
   revision="107">
<author>maggie_n</author>
<date>2008-02-25T20:13:50.097229Z</date>
<msg>My awesome delta number 7.</msg></logentry>
</log>
```

```php
$deltaSvnRevision = $myXml->logentry[0]['revision'];
```

Automagically...

```php
<?php
$runThis = '';

// 1. up or down?
$direction = 'up'; // or down if you want
```

```php
// 2. parse the delta to get DDL + DML
$delta = simple_xml_loadfile('delta7.xml');
if ($direction == 'up') {
    $runThis .= $delta->up;
} else {
    $runThis .= $delta->down;
}
```

schematic

```php
// 3. get delta revision
exec("svn log -r PREV:HEAD delta7.xml --xml", $deltaSvnLog);
$deltaSvnLogAsXml = simple_xml_loadfile($deltaSvnLog);
$deltaSvnRevision = $deltaSvnLogAsXml->logentry[0]['revision'];
```

```php
// 4. what's the package svn version closest to $deltaSvnRevision?
exec("svn log -r PREV:$deltaSvnRevision $pkg --xml", $pkgSvnLog);
$pkgSvnLogAsXml = simple_xml_loadfile($pkgSvnLog);
$pkgSvnRevision = $pkgSvnLogAsXml->logentry[0]['revision'];

exec("svn log -r PREV:$deltaSvnRevision $pkgBody --xml", $pkgBodySvnLog);
$pkgBodySvnLogAsXml = simple_xml_loadfile($pkgBodySvnLog);
$pkgBodySvnRevision = $pkgBodySvnLogAsXml->logentry[0]['revision'];
```

```php
// 5. get contents of the package header and body for each referenced pkg
foreach($delta->packages->package as $pkg) {
    $headerFile = $pkg->header;
    $bodyFile = $pkg->body;

    exec("svn cat -r$pkgSvnRevision $headerFile", $header);
    exec("svn cat -r$pkgBodySvnRevision $bodyFile", $body);

    $runThis .= $header . $body;
}
```

```php
// 6. return contents of header(s) and body(ies) for all referenced packages
return $runThis;


?>
```

Communication between development branches

dev1
(new stuff)

dev2
(new stuff)

qa1
(new stuff)

staging

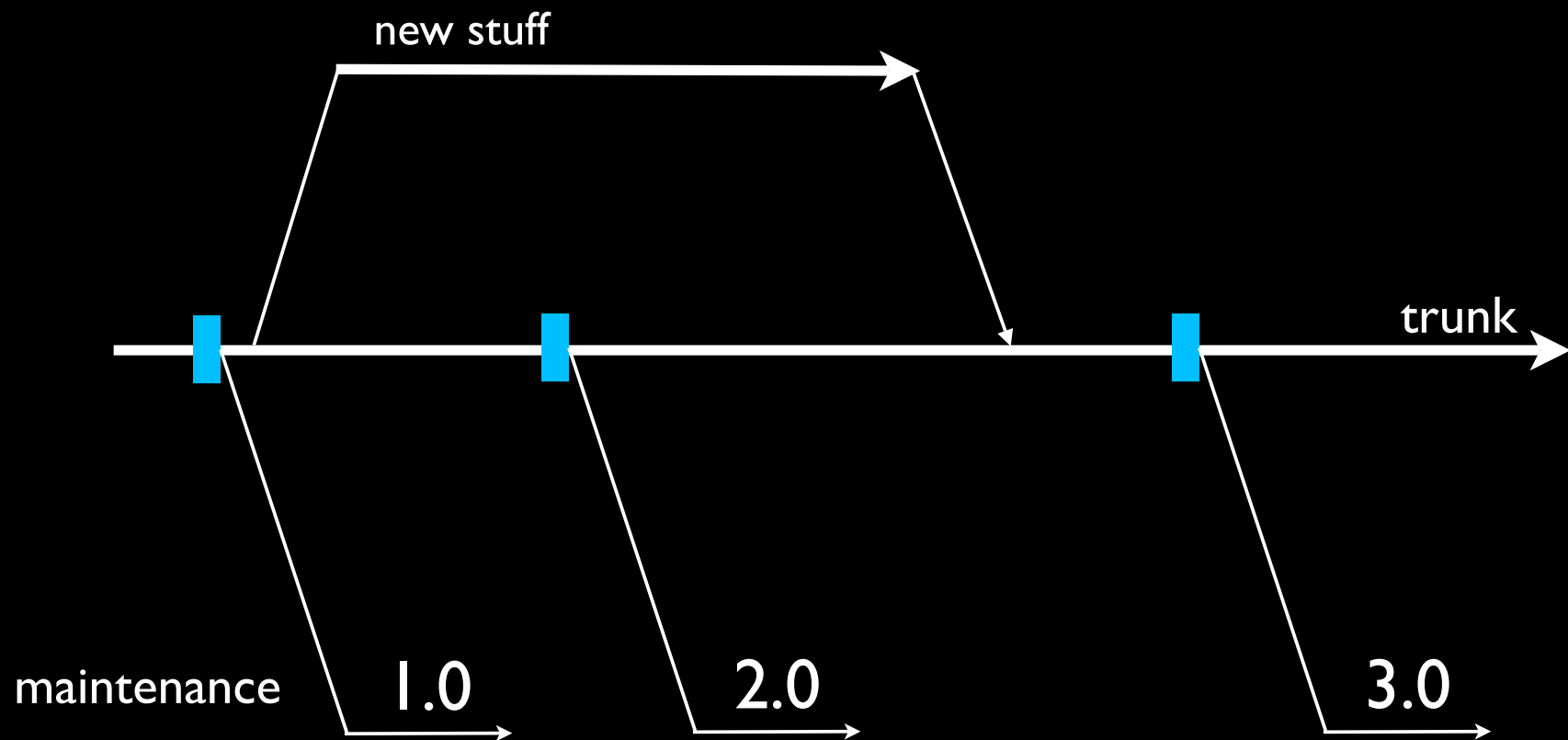prod

dev3
(maintenance)

qa2
(maintenance)

schematic

svn:externals

```
http://svn.example.com/svn/myapp/
    trunk
    branch/
        new_stuff/
        1.0/
        2.0/
        3.0/
    deltas/
```
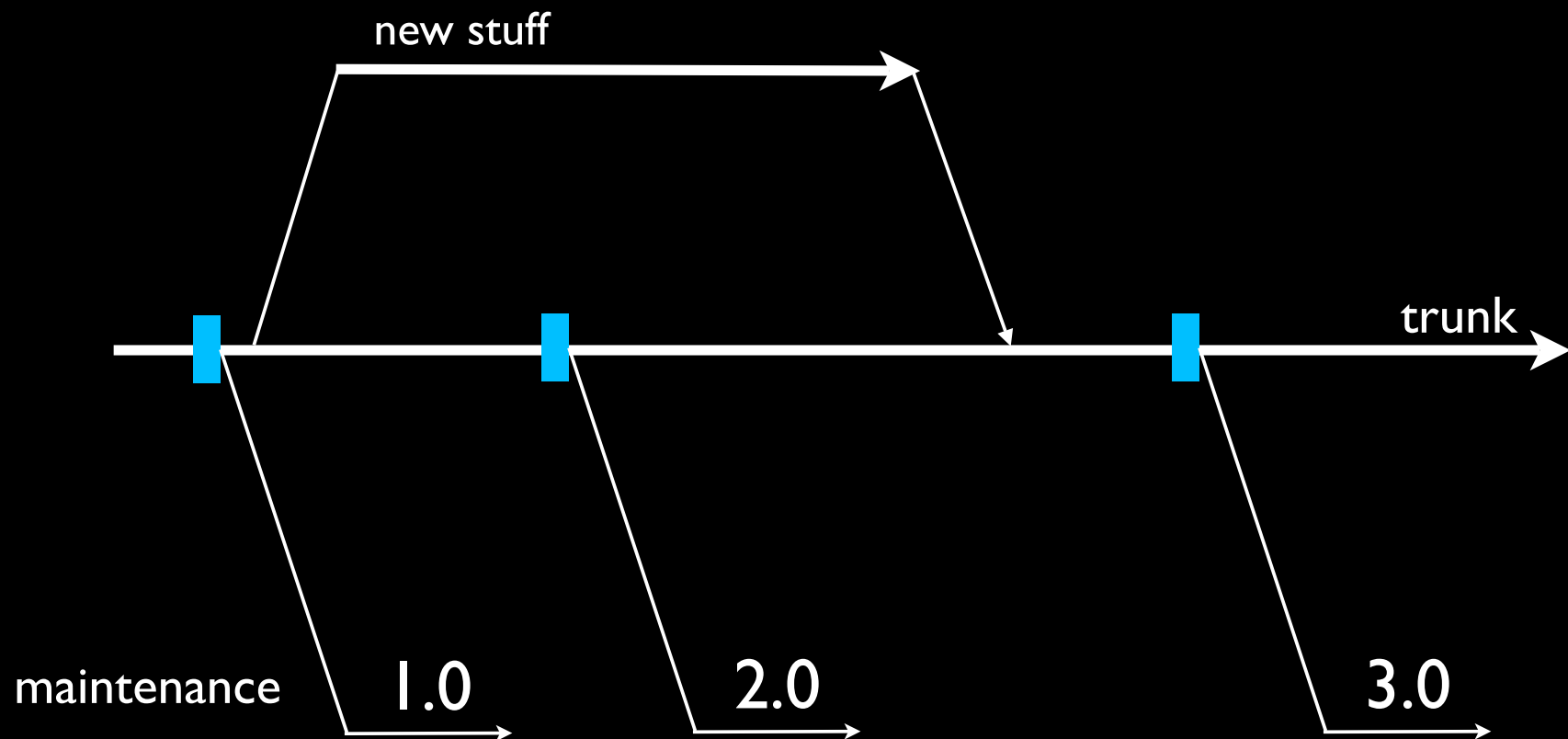
create new deltas directory parallel to trunk and branches

in trunk or branch you're working in:
svn propedit svn:externals .

edit file that opens, add:
deltas http://svn.example.com/myapp/deltas

save and exit

svn up
svn commit

central file to associate delta 2 branch/trunk

trunk:
  - delta1
  - delta2
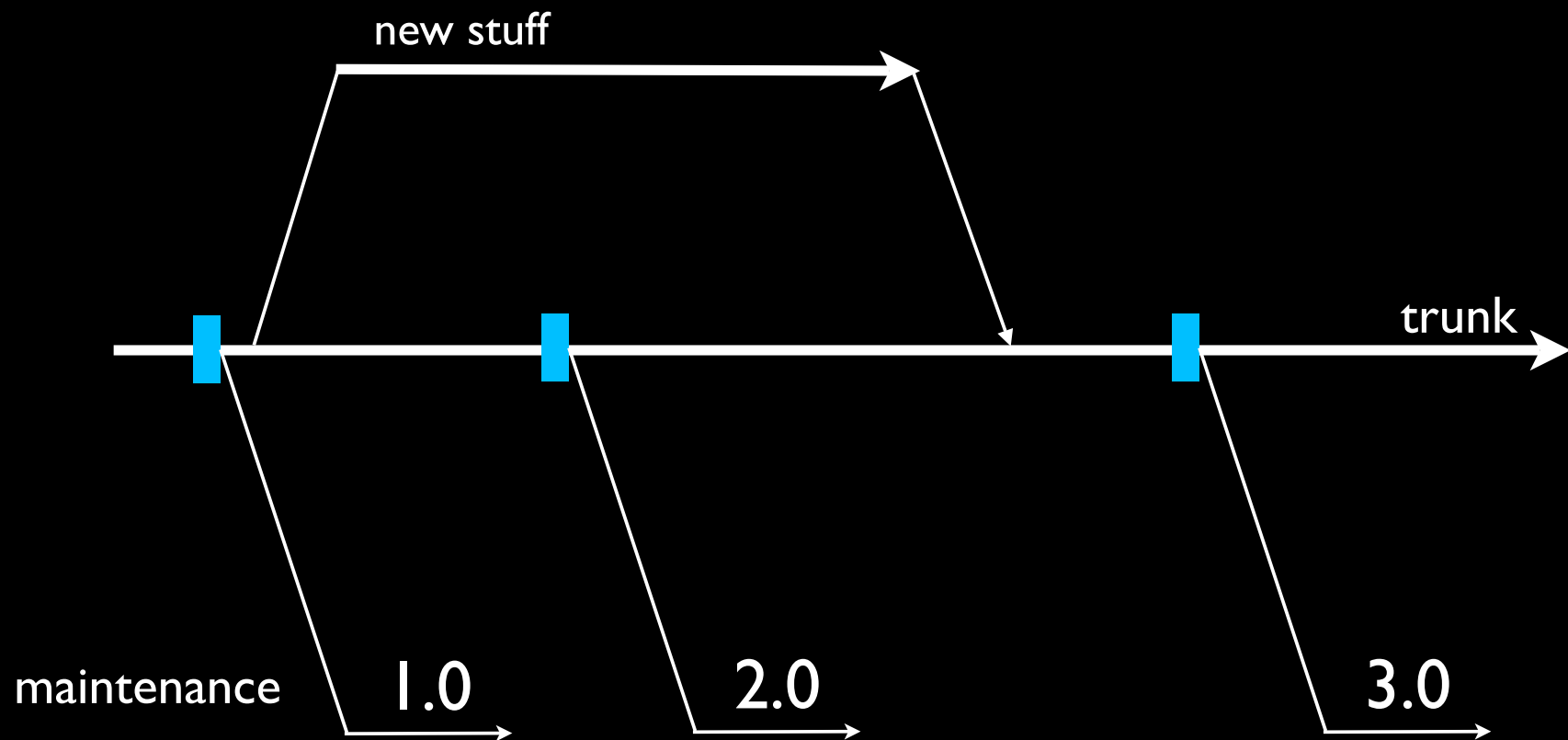
branches/new_stuff
  - delta3
  - delta4

branches/1.0
  - delta1

branches/2.0
  - delta1
  - delta2

branches/3.0
  - delta1
  - delta2
  - delta3
  - delta4

Summary

PHP + DB + XML + svn = awesome!

Questions?

Thanks!

more info:
maggienelson.com