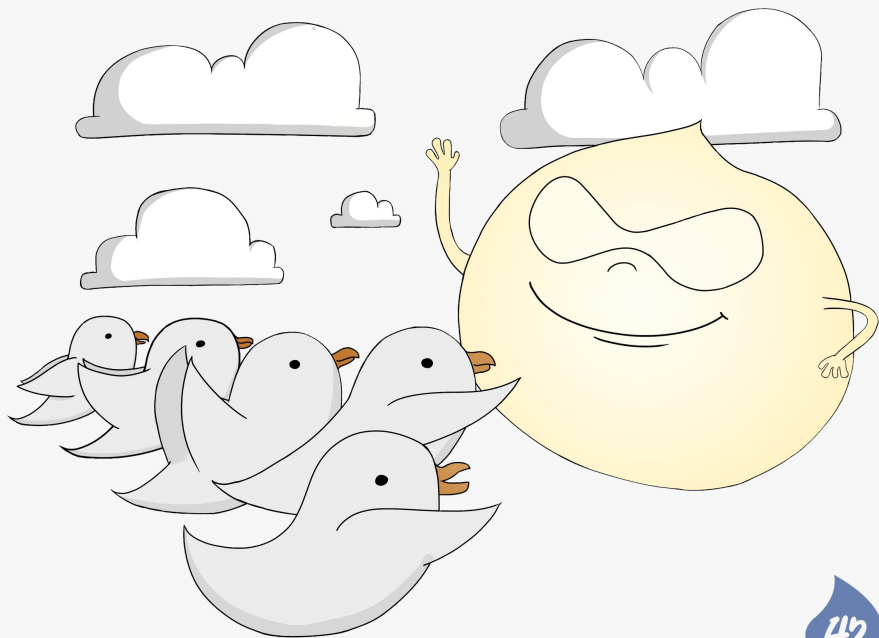


# Planning & Managing Migrations

It's for the birds.  
Har har.



Aimee Degnan /  
aimee@hook42.com

# Me

## Aimee Degnan, Hook 42

- 1996 – Enterprise Web Tech & CMS
- 2006 – PMP, Stanford Advanced PM
- 2008 – Drupal
- 2010 – Certified Scrum Master, Product Owner
- aimee@hook42.com
- @aimeeraed
- [www.hook42.com](http://www.hook42.com)
- @hook42inc



# Hook 42

Full-service digital agency.

Certified Women's Business Enterprise.

20+ years industry experience.

Actively contribute to the community.

- Complex projects
- Process automation
- Drupal
- Migrations
- Multilingual
- SEO



[www.hook42.com](http://www.hook42.com)

@hook42inc

# Who are you?

Project  
Manager?

Product  
Manager?

Developer?

Executive?

**All?!?!?**



# About the session

- There is a lot to cover today.
- We are going to go fast. Enjoy the ride. :)
- Slides are text heavy. Sorry!
- Slides are posted, please stay engaged!
- This information applies to any size migration.

## How?

**Sample project + Background + Phases**

Who has migrated a site?

In one word, describe it. 😊

First, let's get a migration project.

[nexus-travel.com](https://nexus-travel.com)

# Nexus-Travel.com

## Exciting destinations

### Stunning Views In Sydney



### Make This Your New Office



### Explore Egypt by Air



### Tropical Paradise at Reasonable Rates



### Travel to London for Less



### All Trails Lead to Portland



English  
Español  
Français  
Magyar

### Explore with us

There are many exciting destinations in the world! Explore our [trips](#) to learn about these wonderful places and experiences. When you are ready to book your trip, [contact us](#) and we'll make sure you have a vacation of a lifetime.


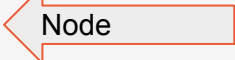
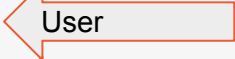
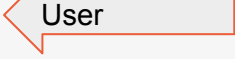
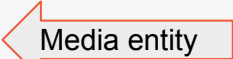
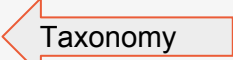

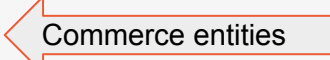
This is taken from the multilingual demo on Drupal.org. See it: [https://www.drupal.org/project/multilingual\\_demo](https://www.drupal.org/project/multilingual_demo)



# Who is Nexus Travel?

- It is an online business that sells pre-planned trips.
- It was built on the now non-supported Drupal 6.
- The website is large.
- It has enterprise grade features.
- There are many types of content on the site.
- Much of their custom code interacts with data.

# About the content

- Locations  Node
- Tours  Node
- Vendors  User
- Members  User
- LOTS of pretty pictures!  Media entity
- Rich content tagging  Taxonomy
- Advertisements are sold to vendors  Blocks
- Commerce (membership, trips)  Commerce entities

# About the project

- Under a tight timeline.
- The “new and improved” features have not been defined.
- Large business investments are dependent on timely release.
- Organic SEO is the largest driver of traffic to their site.

# How do you feel?



answers@hook42.com

Before we start, let's  
understand migration  
projects.

They are easy, right?

# Why migrate?

- Software end of life.
- Mergers and acquisitions.
- Fixing the site is more painful than migration.
- Infrastructure / architecture cleanup.
- Rebranding.
- More...

# Types of migration

- One-to-one. (data + functionality)
- Transformation. (old data → new architecture)
- Multiple sources → single source. (i18n)
- Single source → multiple source. (i18n)

**Real life:**

**Your project may use all types.**

# Frequency of migration

- Single Pass.
- Incremental.

**Real life:**  
**Your project may use both types.**



# Size, scale, and complexity

- Small amount of content.  Manual
- Enough content to invest in migration code.  Program
- Content blob to structured field migration.  Program + Manual

**Real life:**  
**HOW MANY FILES?!!**  
**SOOOOOO MUCH CONTENT!!!**

# Multiple technologies

- Drupal to Drupal
- Flat to Drupal
- Custom DB to Drupal
- Other CMS to Drupal

Thank you, community!





Hmm.....

Hmm.....

Hmm.....

**Real life:**  
**Your project may use many types.**

# Infrastructure considerations

- Pantheon  Can't mv / files.
- Acquia  Files directory structure
- Local hosts vs. remote / shared hosts  Memory, debugging
- Network  Transfer speeds, firewalls

**Real life:**  
**Legitimate impacts to planning.**

# Team considerations

- Projects can be **long**
- Migration may be **after-hours**
- Work is **INCREDIBLY** detail oriented
- Careful, deliberate, correct note-taking is **required**
- Work can be **intense!**

**Real life:**  
**Who likes to work like this?**

# Team specialization

- Migration Project Manager ← Plan and educate
- Source Technology Engineer ← Access source data
- Migration Engineer ← Develop migration code
- Migrator ← Run migrations / recover from failure
- Data Specialist ← Test the migrated data

**Real life:**  
**Where do you get these people?**

# Role-specific considerations

- Business owners
- Account managers
- Project managers
- Migration engineers
- Developers
- Site builders
- Themers
- And more...

Make it **easier** on your team.

**Simplify** where you can.

# Spreadsheets!

No cell left behind.



No, really.  
Spreadsheets.

Migrations have a  
lot of moving parts.



*Details  
Shmetails.*

# Why not a bug tracker?

A spreadsheet is a custom DB table(s) w/ all variables.



**Thorough planning**  
and  
**vigilant management**  
leads  
to  
**project success.**

And the numbers prove it.

**Let's do the math!**

Start to get started.

# Agile vs. Waterfall?

There are benefits of both methodologies.

## Waterfall:

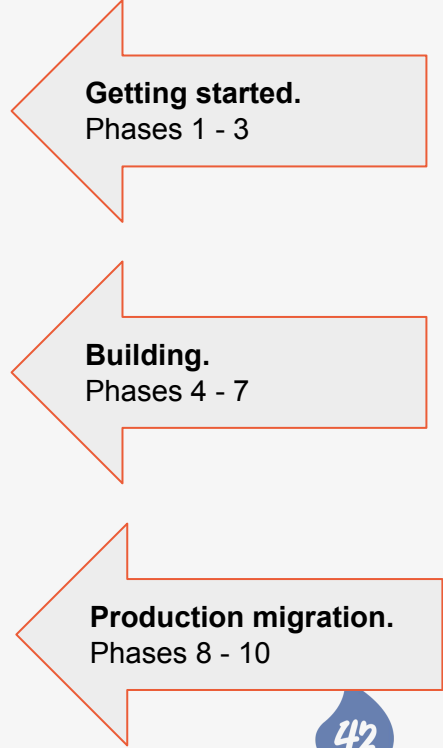
- Order of operations.
- **Sign-off** and commitment.

## Agile:

- Culture supports adjustments for new information.
- Meeting, reporting, review, and **acceptance** cadence.

# Phases of a migration project

1. Pre-project education
2. Audit for migration
3. Discovery
4. Architect the new site
5. Migration mapping
6. Development phase
7. Pre-production migration
8. Site testing and migration audit
9. Go live!!!!!!!!
10. Post-launch validation



**Getting started.**  
Phases 1 - 3

**Building.**  
Phases 4 - 7

**Production migration.**  
Phases 8 - 10



## ◆ 1) Nexus-Travel.com Migration

### ▼ 2) Getting Started

- 2.1) Phase 1: Project Kickoff / Education
- 2.2) Phase 2: Audit for Migration
- 2.3) Phase 3: Discovery

### ▼ 3) Building Stuff

- 3.1) Phase 4: Architect new site
- 3.2) Phase 5: Migration mapping

#### ▼ 3.3) Phase 6: Development

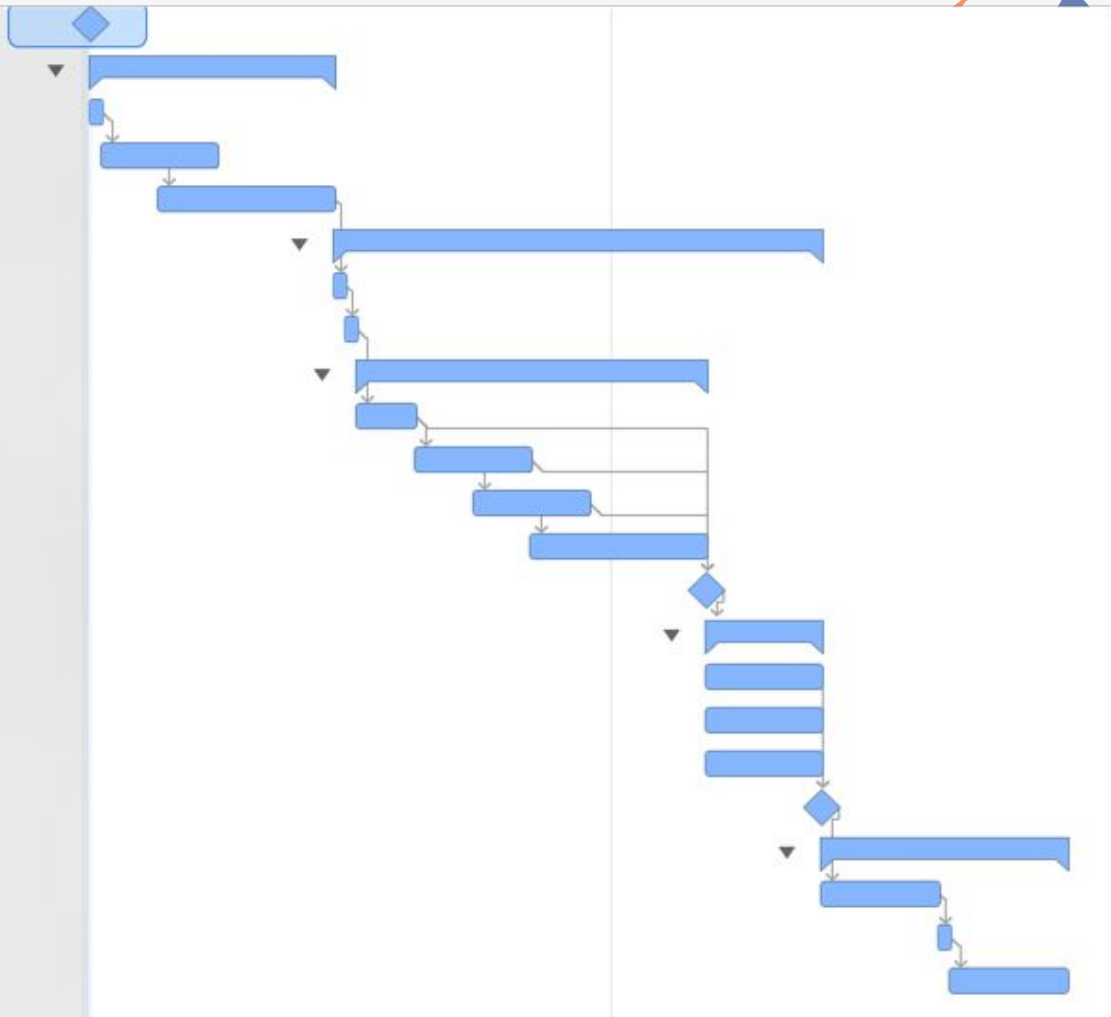
- 3.3.1) Infrastructure
- 3.3.2) Site building
- 3.3.3) Development migration passes
- 3.3.4) Theming
- ◆ 3.3.5) Development complete

#### ▼ 3.4) Phase 7: Pre-production migrations

- 3.4.1) Testing (Functional & Data)
- 3.4.2) Bug fixes
- 3.4.3) Incremental migration passes
- ◆ 3.4.4) Development and bug fixing complete.

### ▼ 4) Production Migration

- 4.1) Phase 8: Site testing & migration audit
- 4.2) Phase 9: Go live!
- 4.3) Phase 10: Post-launch validation





Getting started.

# Phases of a migration project

1. **Pre-project education**
2. **Audit for migration**
3. **Discovery**
4. Architect the new site
5. Migration mapping
6. Development phase
7. Pre-production migration
8. Site testing and migration audit
9. Go live!!!!!!!!
10. Post-launch validation



**Getting started.**  
Phases 1 - 3

# 1. Pre-project education

## Goals:

- Set expectations of project activities.
- Clarify the impact of requirements freeze.
- Identify possible phased statements of work.

**Real life:**

**It is ready when it is ready.**

# 1. Pre-project education

## Migration projects take:

- Time
- Specialization
- Requirements lockdown
- Project fitness
- Transparency

# 1. Pre-project education

## Nexus Travel:

- **Undefined** new features are a **risk to schedule**.
- **Aggressive schedule** may **impact** developer **work-life balance** because of nights and weekend work.
- **Mitigate** customer **expectations** with **new launch dates**.

## 2. Audit for migration

### Goals:

- Surface the As-Is details of the current site(s)
- Begin understanding data
- Familiarity with site functionality
- Re-educate the business with findings

## 2. Audit for migration

### Artifacts:

- Risks register
- Content audit (structure, data, size, source)
- Functionality audit (surface custom code!)
- Data health audit
- Infrastructure audit
- Functionality specific audits: SEO, Accessibility, Access
- Source URL lists (url patterns, special pages)
- Links to representative content. Everyone uses them!

## 2. Audit for migration

### Artifacts:

- Risks register
- Content audit (structure, data, size, source)
- Functionality audit (surface custom code!)
- Data health audit
- Infrastructure audit
- Functionality specific audits: SEO, Accessibility, Access
- Source URL lists (url patterns, special pages)
- Links to representative content. Everyone uses them!



## 2. Audit for migration

### Lessons learned:

- Very few developers know how to audit for migration.
- Takes longer than you'd expect, even using tools.
- Auditing twice is costly.
- Do it right the first time.
- **No cell left behind!! Blank != N/A**
- Keep your artifacts and info in one place.
- Mitigates: "Oh, I didn't think about that."

Name / Feature	Entity type	Source Complexity	Count
Members	user	profile fields, multiple roles	50,000
Lots of pretty pictures	files + media	Disorganized, bad file names	60,000 / 65,000
Many vocabularies	taxonomy	Heavily tagged content	20
Many terms	taxonomy	Heavily tagged content	800
Basic Page	node		200
Locations	node	each: 5 pictures, 150 fields, node hierarchy, many specialized fields - geo location	3,000
Vendors	node	specialized users + roles + permissions, media, locations	300
Trips	node	150 fields, many relationships	15,000
Ads	blocks		1,000
Share Your Trip	node	multiple pictures and videos	5,000
Commerce	commerce entities		2,325,000
Aliases / Redirects	alias / redirect		720,320 / 1,440,640

# 3. Discovery

## Goals:

- Define new functionality and improvements.
- Prioritize feature development, with data in mind.
- Capture expectations of data on migration.
- Re-educate the business with findings.

## Tools:

- Leverage the spreadsheets started by the audit!

# 3. Discovery

## Artifacts:

- Feature list.
- Feature requirements.
- Project glossary with AKAs.
- Elaborate on the representative links list.

# 3. Discovery

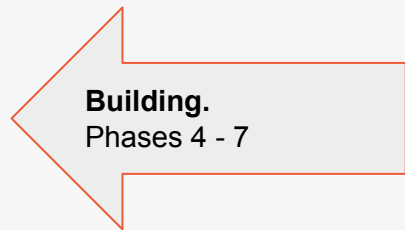
## Nexus Travel:

- Keep the old data.
- Transform select lists into taxonomy terms.
- Content team wants to make new content for “Paid Landing Pages” before site go-live.
- “Most functionality is the same.” Don’t trust this statement until sign-off for development.

Building stuff.

# Phases of a migration project

1. Pre-project education
2. Audit for migration
3. Discovery Phase
- 4. Architect the new site**
- 5. Migration mapping**
- 6. Development phase**
- 7. Pre-production migration**
8. Site testing and migration audit
9. Go live!!!!!!!
10. Post-launch validation



# 4. Architect the new site

## Goals:

- Define new content structures.
- Define infrastructure with migration considerations.



# 4. Architect the new site

## Artifacts:

- Leverage the spreadsheets started by the audit!
- Feature development roadmap
- Site architecture spreadsheet
- URL pattern planning

# 4. Architect the new site

## Lessons learned:

- Every **ENTITY** is a migration pass! *(pssst. paragraphs)*
- **Media** entities require **at least two passes** *(files + entity)*
- You **MUST** architect **EVERYTHING** before building.
- **DO NOT** let your site builders just build without writing it down. **Ever.**

## 4. Architect the new site

### **Nexus Travel:**

- Architects want to use Paragraphs for layout.
- Commerce does things in a new way to support older functionality.
- Business is getting excited and adds more requirements to the Paid Landing Page feature, add A/B testing, and add conversion funnel analytics.

# 5. Migration mapping

## Goals:

- Document migration expectations for the business.
- Provides detailed requirements to Migration Engineers.
- Creates a testing matrix for post-migration data audit.

# 5. Migration mapping

## Artifacts:

- Leverage the spreadsheets started by the audit!
- Migration mapping spreadsheet:
  - Source → destination fields + transformation
  - Taxonomy term / select list → term mapping
  - Migration dependencies / migration order

# 5. Migration mapping

## Lessons learned:

- Done in parallel with new architecture.
- Mind the finer data details:
  - Types, field length, formats, dates, and filters!
  - Select lists → taxonomy / Term → term
- Splitting blobs (the Body) → structured content take extra programming and data testing.

# 5. Migration mapping



## Nexus Travel:

- Basic page “Tags” select list has a text string mapping that splits to many different terms in new structured vocabularies.
- Image field is mapped to a specific media entity type.
- Hey, let’s add Spanish!

# 6. Development phase

## Goals:

- Get 'er done.
- Development of EVERYTHING!!!!
- Reestimate the work, if necessary.
- Go back to site architecture phase, if necessary.
- Reeducate the client.



# 6. Development phase

## Artifacts:

- The site.
- Migration code.
- Infrastructure setup.
- Detailed site rollback process.
- Go-live checklist: full list of migrations, duration, expected behaviors.

## 6. Development phase

### Considerations:

- Site building MUST be complete before migration development starts.\*
- Create the migration dependency / order before code.
- Develop migration code.
- Developer is responsible for first population of go-live checklist.
- Don't over engineer. You are only doing this once.\*

## 6. Development phase

### Lessons Learned:

- Max joins on MySQL DB is 61.
- Documentation is your friend.
- Comments / UGC migrations need the parent entity!
- Watch the published / unpublished status of source.
- DOM parsing leads to memory leaks.
- Splitting a body field to structured field? Good luck!

## 6. Development phase

### **Nexus Travel:**

- The Trips content migration hit the 61 join limit.
- The Share My Trip migration ran out of memory and had to be batched in groups of 1,000 in each pass.
- The network latency between one of the developers homes is really high and bandwidth is low and can skew migration run-time.

Name / Feature	Full Migration Time	Developer Migration Notes	Count
Members	10 min - 90 min*		50,000
Lots of pretty pictures	16 hours*	file copy down from source + file cleanup + file copy up to destination + number of gigs/internet speed from both ends	60,000 / 65,000
Many vocabularies	2 min	Heavily tagged content	20
Many terms	4 min	Heavily tagged content	800
Basic Page	2 min		200
Locations		each: 5 pictures, 150 fields, node hierarchy, many specialized fields - geo location	3,000
Vendors		specialized users + roles + permissions, media, locations	300
Trips	60 min - 120 min	lots of joins!	15,000
Ads	blocks		1,000
Share Your Trip	250 min*	memory leak, run in batches	5,000
Commerce			2,325,000
Aliases / Redirects			720,320 / 1,440,640

*hook*

42

42

# 7. Pre-production migrations

## Goals:

- Keep running migrations
- Debug and test data (dev team + engaged client)
- Populate the bulk of the data
- Estimate duration of final, go-live migrations.

## Artifacts:

- Leverage the spreadsheets started by the audit!
- Go-live checklist: track time, success / failure, issues.

# 7. Pre-production migrations

## Nexus Travel:

- We have to add X, Y, and Z to the Trips migration!

## What happens now?

- If anything new, iterate on phase 4 - 7 over and over.
- If any change impacts a related migration pass, you have to rollback + run other related migrations.
- **Migrate ++, Cost ++, Time ++**



## ◆ 1) Nexus-Travel.com Migration

### ▼ 2) Getting Started

- 2.1) Phase 1: Project Kickoff / Education
- 2.2) Phase 2: Audit for Migration
- 2.3) Phase 3: Discovery

### ▼ 3) Building Stuff

- 3.1) Phase 4: Architect new site
- 3.2) Phase 5: Migration mapping

#### ▼ 3.3) Phase 6: Development

- 3.3.1) Infrastructure
- 3.3.2) Site building
- 3.3.3) Development migration passes
- 3.3.4) Theming

#### ◆ 3.3.5) Development complete

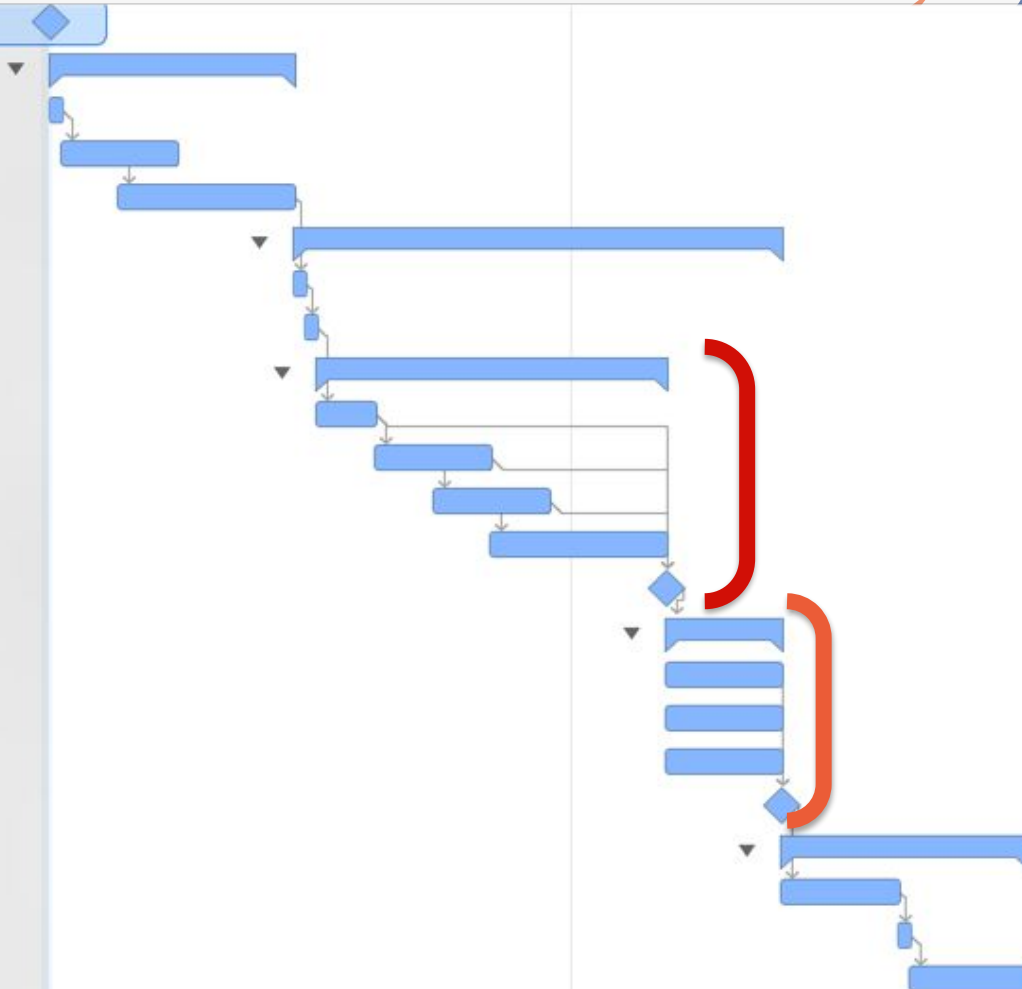
#### ▼ 3.4) Phase 7: Pre-production migrations

- 3.4.1) Testing (Functional & Data)
- 3.4.2) Bug fixes
- 3.4.3) Incremental migration passes

#### ◆ 3.4.4) Development and bug fixing complete.

### ▼ 4) Production Migration

- 4.1) Phase 8: Site testing & migration audit
- 4.2) Phase 9: Go live!
- 4.3) Phase 10: Post-launch validation





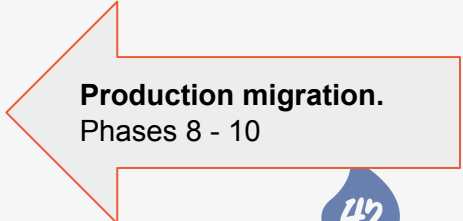
# Do the math...

- **PER MIGRATION**
  - 2 hours of definition
  - 8 hours of development
  - 120 min of migration x 4 rounds of testing (developer, client, migrator, data/site qa)
  - 2 hours deployment overhead
- **Total coverage hours:** 20
- **Total coverage cost:** \$2,000 - \$4,000 (rate variance)
- **Total coverage time:** Addition of 1 calendar week

Production migration.

# Phases of a migration project

1. Pre-project education
2. Audit for migration
3. Discovery
4. Architect the new site
5. Migration mapping
6. Development phase
7. Pre-production migration
- 8. Site testing and migration audit**
- 9. Go live!!!!!!!**
- 10. Post-launch validation**



**Production migration.**  
Phases 8 - 10

# 8. Site testing and data audit

## Goals:

- Test new site architecture with migrated data.
- Polish layout and functionality / fix bugs.
- Add additional, manual or new content.

# 8. Site testing and data audit

## Artifacts:

- Leverage the spreadsheets started by the audit!
- Go-live checklist: track time and issues
- Browser testing
- SEO testing / redirects
- Performance tuning
- Go-live preparation

## 8. Site testing and data audit

### Lessons learned:

- There is a “**Moment of Truth**” when the new, to-be production server becomes “**Non-live Production**” and the migrations are “**REAL**”.
- **Rollback can be painful**, time consuming, and require your pre-allocated developer resources.
- **Do not run migrations** in your non-live production environment **until they have passed testing**.

# 8. Site testing and data audit

## Nexus Travel:

- We have to add A, B, and C to Share Your Trip data!
- This is going to delay launch, send an email to vendors!

## What happens now?

- If anything new, iterate on phase 4 - 7 over and over.
- If any change impacts a related migration pass, you have to rollback + run other migrations.
- **Migrate ++, Cost ++, Time ++**

# Do the math again...

- **PER MIGRATION**
  - 2 hours of definition
  - 8 hours of development
  - 120 min of migration x 4 rounds of testing (developer, client, migrator, data/site qa)
  - 2 hours deployment overhead
- 3 migrations impacted
- **Total overage hours:** 60
- **Total overage cost:** \$12,000 - \$24,000 (rate variance)
- **Total overage time:** Add 2 calendar weeks



## 9. Go-live!

### Goals:

- Final migration & smooth cutover.

### Artifacts:

- Go-live checklist. It isn't just migration passes.
- DNS DNS DNS DNS DNS

# 9. Go-live!

## Lessons Learned:

- Practice migrations before cutover.
- Practice your roll back before cutover.
- You and your team will probably be tired.
- This your “A” game.
- Relax. Grab a glass of wine.
- Something is going to happen.

# 9. Go-live!

## **Nexus Travel:**

- The hosting DNS failed.
- Source and new-prod server failed for 4 hours.
- Failure was identified on one of the backups, not a migration pass. Whew!
- Launch took 12 hours vs. the 8 due to the outage.
- The DNS propagated in a timely manner.

# 10. Post-launch validation

## Goals:

- Did it work?
- Did we miss something on cutover?

## Lessons Learned:

- This phase is important.
- You aren't done when the site is cutover.

# 10. Post-launch validation

## Artifacts:

- Speed tests.
- SEO tests.
- Error logs.
- Feedback from site users.

# 10. Post-launch validation

## **Nexus Travel:**

- The 404 logs showed some missing redirects.
- Vendors were happy with their new features.
- Some data expected by members was “missing”.

# Takeaways

- Incomplete requirements = rework = increased time and costs.
- Migration work is exponentially longer due to the nature of development and testing.
- Your team may change over time.  
Write everything down!

