Införande av Koha vid Lunds universitetsbibliotek, en projektrapport

-Lund, 2018

Bild: Dave Sherohman

I tidernas begynnelse fanns ett kungarike. Det var ett kungarike under ständig förändring, och av detta finns enbart rester kvar. Dev-klonernas folk har arbetat hårt för utveckling, samtidigt som en drake har lurat. Men ännu råder inte frid i riket; Galera-triumviratet håller på att störtas, och ryktet går om att en mästare och dess slavar ska ta över…

Innehåll

Abstract

Det här är en rapport över IT-avdelningens på Lunds universitetsbibliotek erfarenheter över att lansera ett nytt bibliotekssystem. Vi behövde, på grund av kostnader och av konkurrensutsättningskrav, byta bibliotekssystem. Valet föll på den öppna programvaran Koha och förberedelsearbetet påbörjades i juni 2017, för lansering ett år senare. Rapporten beskriver i ganska stor detalj arbetet under detta år, med vår metod - scrum, hur vi arbetar, vilken utveckling vi har gjort, våra erfarenheter av projektet, samarbete med andra som använder Koha, och lite kort om hur vi ser på framtiden.

Rapporten är skriven av Stina Hallin, med hjälp av Dave Sherohman, David Holoshka och Snorri Briem.

Bakgrund och förarbete

Om Lunds universitets bibliotek

Lunds universitets bibliotek består av 26 stycken bibliotek, fördelade på 8 fakulteter, samt universitetsbiblioteket och två särskilda verksamheter. De 26 biblioteken servar sina olika fakulteter och tillhandahåller kurslitteratur och specialkompetens i de respektive ämnena. Universitetsbiblioteket är ett pliktbibliotek, och lånar dessutom ut materialet utöver uppgiften att bevara en kopia av allt tryckt material i Sverige. Vi har cirka 400 000 utlån per år, vi har cirka 30 000 aktiva låntagare, vi har över 2 miljoner bibliografiska poster i vår Opac, det arbetar runt 250 personer i nätverket.

Varför byta bibliotekssystem?

Vi har haft samma leverantör av bibliotekssystem sedan slutet av 1980-talet. De har levererat olika system; det senaste infördes 2001. Det var alltså hög tid för lite konkurrensutsättning. Virtua, som vi använde, köptes för några år sedan upp av Innovative, och vidareutvecklingen av vårt system skulle upphöra framöver, så att Innovative kunde satsa på nya, moderna system. Kostnaden för systemet, inklusive databaslicensen, hade ökat, och driften låg på LUs datacentral, vilket medförde ganska höga avgifter för vårt bibliotekssystem.

Att det skulle bli ett Open source system som skulle ersätta vårt gamla kommersiella system var ingen självklarhet. Vi tittade också på olika proprietära system, men eftersom vi enbart skulle byta vårt ILS, blev Koha det naturliga valet, då det vid tidpunkten för bytet inte fanns något nytt, modernt system som enbart var ett ILS. De andra systemen erbjöd istället totallösningar, vilket vi då inte var intresserade av.

Beslutet

I januari 2017 fick IT-avdelningen uppgiften att testa Koha i vår egen miljö. Vi fick kort om tid; i april skulle resultatet in, för att beslut i Lunds universitets biblioteks ledningsgrupp skulle kunna tas i början av maj. Under denna period samlade vi in processer från våra olika bibliotek. Dessa processer testades sedan av i Koha för att se om det vi gjorde i vårt gamla system, Virtua, fungerade att göra i Koha. Vi fann att det mesta fungerade likadant, eller på ett någorlunda liknande sätt, och att en del funktionalitet saknades. Våra resultat samlades in och en arbetsgrupp bestående av bibliotekarier från några olika fakulteter skrev ett beslutsunderlag som LUBs ledningsgrupp sedan beslutade efter. I juni 2017 började arbetet på allvar med att arbeta med Koha, och vi fick ett år på oss att bli klara för lansering; sommaren 2018 skulle vi lansera ett nytt bibliotekssystem för Lunds universitets bibliotek.

Projektorganisation

Projektgruppen som arbetade med införandet av Koha vid LUB bestod av IT-avdelningen (7 personer), en katalogspecialist, en testansvarig, samt fyra resurspersoner som arbetat med allt från utveckling och översättning till utbildning och marknadsföring. IT-avdelningen består av programmerare, systemutvecklare och bibliotekarier. Ingen har på pappret arbetat 100% med Koha, även om det under vissa perioder blivit så. Utöver arbetet med Koha hade alla sina ordinarie arbetsuppgifter, även om dessa stundtals behandlats synnerligen styvmoderligt. För att förenkla prioriteringen, vilket inte är lätt när man arbetar på en IT-avdelning och ska serva och hjälpa hela personalstyrkan på universitetsbiblioteket med sina respektive system, fick vi sätta en gräns på att maximalt lägga 20% på övriga arbetsuppgifter. Det lyckades de flesta hålla den mesta av tiden. De personer som inte tillhörde IT-avdelningen har haft andra överenskommelser med sina respektive chefer, och bidragit i varierande grad.

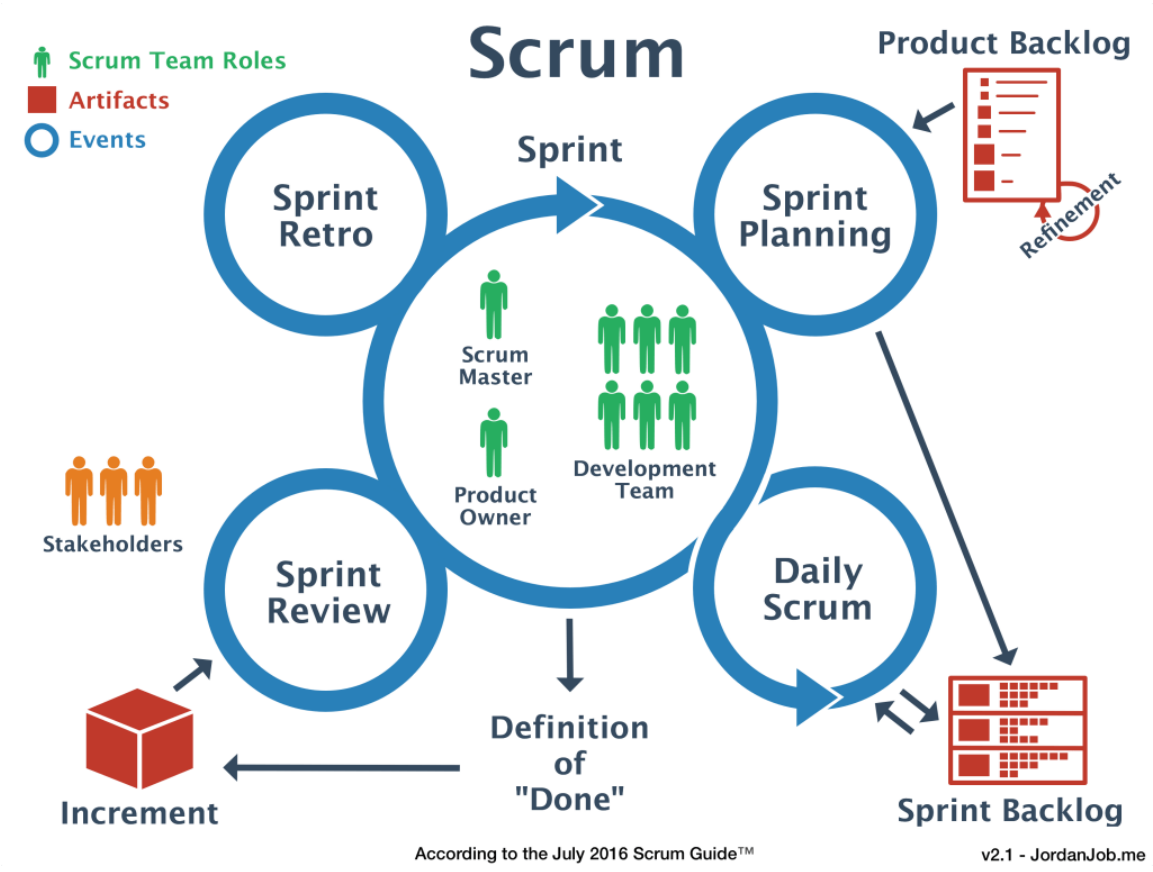
Metod

Strax innan projektet med Koha drog igång, beslöt IT-avdelningen sig för att börja arbeta agilt. Scrum-metoden valdes, och alla på avdelningen läste in sig på metoden. Det praktiska arbetet med scrum började när projektet på allvar drog igång. Antagligen hade det varit klokt att ha använt metoden i praktiken före ett så stort och tidspressat projekt som införande av Koha innebär. I Lund har vi använt ett projektverktyg som heter Redmine. Till det finns en scrum-plugin, och den har varit till stor hjälp. Redmine är fri programvara, men det finns möjlighet att betala för att få specifik utveckling utförd. Programmet har fungerat bra för våra behov, även om vi ibland saknat viss funktionalitet.

Vi använder oss av 2-veckors sprintar. Varje sprint, som löper mellan måndag och fredag i nästföljande vecka, inleds av ett planeringsmöte. Det mötet tar ca 2 h. Produktägaren och scrummasters har förberett sprintarna, så arbetet som görs på planeringsmötet är att bestämma gemensamma mål för sprinten, bedöma tidsåtgång för respektive uppgift och även att fördela arbetsuppgifterna. Vi skriver också tasks (arbetsuppgifter), om det inte redan finns. Därefter löper sprinten på, med dagliga scrummöten. Dessa brukar vara cirka 10 minuter, ibland tappar vi disciplinen och pratar för länge, ambitionen är dock att det ska vara korta, stående, möten. På dessa möten går vi igenom sprintboarden, och alla får redogöra för sitt arbete helt kort. Om någon har

något som hindrar eller försvårar arbetet försöker vi lösa det där. Sprinten avslutas med en review och en retrospective. Reviewerna var under andra halvan av projektet publika. För oss innebar det att vi hade en stående tid varannan fredag, där den som ville fick komma för att se när vi visade upp det arbete som gjorts i sprinten. Ibland hade vi lite att visa, andra gånger fick vi begränsa oss för vi ville hålla dessa möten till max en timme. Dessa publika reviewer har varit till stor hjälp, då det gav tillfälle för folk att fråga och ifrågasätta vårt arbete, och vi kunde snabbare fånga upp om det var något vi hade tänkt fel kring. Vi fick också tillfälle att visa vårt kommande biblioteksystem, och fick många frågor om sådant vi egentligen inte hade visat. Det var också ett bra tillfälle, för oss att smyga in lite utbildning, och för bibliotekarierna att vänja sig vid ett nytt system. Den första halvan av projektet hade vi mer interna reviewer, där vi högg tag i folk samma dag, och bad dem titta på vad vi hade gjort. Det gav också input, men diskussionerna i de något större publika reviewerna, var mycket värdefulla, och hjälpte arbetet framåt.

Retrospective-möten som vi har varit noga med att också ha i slutet av varje sprint, gav väldigt mycket i början av projektet. Dessa möten har förändrats under vägen, men det de har gemensamt är att vi har utvärderat vårt arbete under dessa möten. Ibland har vi skrivit på post-it-lappar, till exempel 3 bra saker och 3 saker som kan förbättras. Det vi utvärderar är alltså arbetet med den senaste sprinten. Ibland har vi tänkt själv, ibland har vi diskuterat i smågrupper. Efter några gånger blev det tydligt vad som var bra respektive mindre bra, och vi försökte på så vis lära oss att bli bättre. Vi har också lärt oss att mat eller fika förenar, och på sistone har vi pratat om projektet (och annat) på olika lunchrestauranger i stan. Då har vi också lärt oss var det finns godast hamburgare i Lund. Precis i början då vårt agila arbete började, bjöd vi en en projektledare som arbetar med scrum. Hon poängterade vikten av att ha retrospective-möten, och det är vi glada för nu, det är ett givande inslag, så om man känner sig lockad att prioritera bort ännu ett möte och tar bort retrospective, gör man sig själv en björntjänst.



Samarbete

Under perioden för projektet gick Göteborgs universitetsbibliotek live med Koha. Innan dess hade ett par andra större universitets-/högskolebibliotek gått över. Stora bibliotek med Koha fanns också i Danmark och Norge. Att ha andra bibliotek att fråga om allmänna eller specifika saker, har varit värdefullt. Med tidigare system har LUB haft samarbete med Göteborg och Uppsala, då vi haft samma system under samma tid. Därför kändes det bra att kunna fortsätta systemsamarbete med andra bibliotek i liknande situation. Vi har bytt kod och erfarenheter med framförallt Göteborgs UB, och det har varit väldigt givande.

För att lära oss själva mer, och för att knyta kontakter, anordnade vi även en hackfest i Lund vintern 2017, och det var givande även om det inte blev så mycket hackande just då. Ämnen som berördes var till exempel Testing/QA, infrastruktur och work flows samt Elastic search. Vi bjöd in via Slackkanalen och vi var 18 deltagare från Sverige och Norge[1]. Bara att byta erfarenheter och höra vad andra gör och tänker tar verksamheten vidare. De digitala mötena som ordnats under 2018 har också varit bra, som en kanal för erfarenhetsutbyte.

1 https://koha.se/koha-i-sverige/summary-of-the-first-swedish-koha-user-group-hackfest-in-lund/

Projektet

Processkartläggningen och testerna under våren 2017 hade gett oss ett bra underlag för vårt vidare arbete, och vi visste ungefär vad vi behövde utveckla själva och vad som kom med Koha. Ingen av oss hade tidigare erfarenhet i större omfattning av Koha, och säkert gjordes en del felbedömningar på grund av förutfattade meningar om vad funktionalitet egentligen innebär och vad man kan förvänta sig av de olika konfigurationsmöjligheterna.

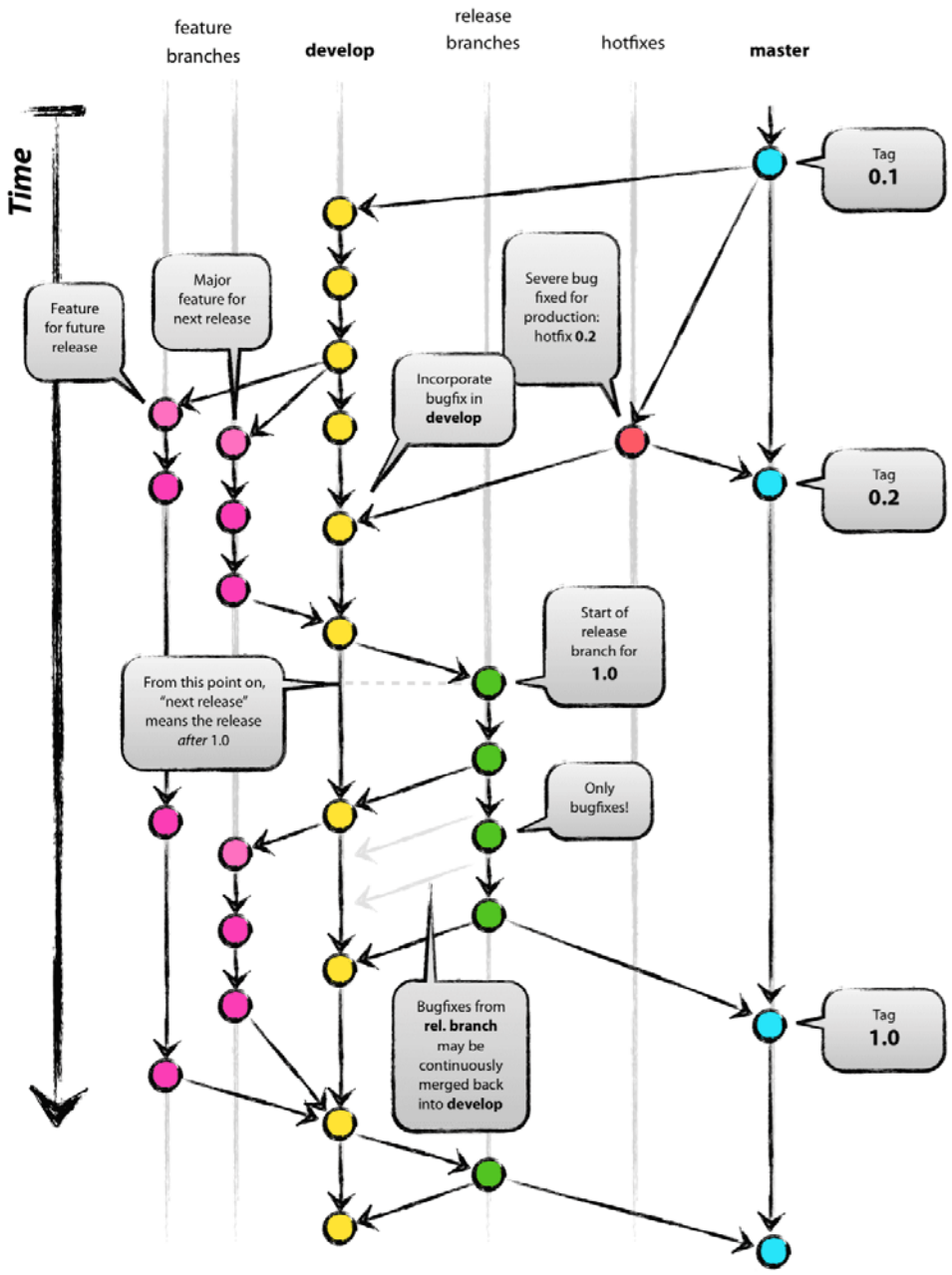
Git

Vi har använt Git för att hantera vår egen källkod, så det föll sig naturligt att använda Git även för att hantera Koha. För att få bättre överblick och för att lättare tilldela rättigheter till personer bestämde vi att installera Gitlab, som är en Git repository manager för hantering av öppen källkod med många användbara funktioner.

Vi började med att klona den senaste stabila community versionen och la den på vårt repository. Vi har sedan fortsatt att arbeta med den, med en välprövad utvecklingsmetod (se bild nedan). I metoden finns det två grenar som lever för evigt: master och dev (ibland kallad devel). Mastern är huvudgrenen och innehåller kod som alltid ska vara redo för produktion. Dev-grenen innehåller de senaste utvecklingsändringarna för nästa release. För att vi ska kunna arbeta parallellt i teamet, och även kunna följa utveckling och buggfixar använder vi supportgrenar som forkas från dev. Dessa grenar har, till skillnad från master och dev, en begränsad livslängd. Oftast korresponderar de till en specifik uppgift från Redmine (sprintboarden). När väl koden från en supportgren är verifierad att fungera som tänkt, mergas den tillbaka in i dev-grenen.

När dev-grenen är stabil flyttar vi den till en release-gren. Koden i release-grenen testas sedan i en produktionsliknande miljö. Vi ändrar enbart i release-grenen om det gäller buggar som hittades vid testningen. Efter tester och eventuella rättningar mergar vi den till master-grenen som en ny release. När vi hittar allvarliga buggar i produktion som behöver åtgärdas snarast, använder vi oss av hotfixar; genom en hotfix-gren från mastern som testas och sedan mergas tillbaka in i mastern. Hotfixar ska alltid granskas av någon annan innan den får mergas in i produktionsmiljön.

De flesta i projektteamet har haft egna utvecklingskloner. Den utveckling, förändringar eller egna tester vi gjort, har vi utfört på våra egna kloner. Om vi haft en uppgift i sprinten att till exempel utföra någon kodändring, översättning eller annat serverarbete, har den först gjorts på den egna klonen. När arbetet verifierats skickas det till dev via gitlab.



Från Vincent Driessens nvie-blogg.

Migrering - från Virtua till Koha

Arbetet med att flytta och konvertera data från det gamla systemet Virtua till det nya Koha, påbörjades cirka ett år före lanseringen av systemet. En undersökning av Kohas importverktyg av marc visade på att det inte var avsett för import av stora eller ens medelstora marcfiler. Våra 2.4 miljoner exemplarposter fick programmet att krascha. Vi behövde alltså utveckla egen kod för att migrera vår data till Koha.

Vi hade lite samarbete med Göteborgs universitetsbibliotek som var upptagna med att få ut sin egen installation av Koha i tid. De var ändå snälla nog att erbjuda en del av sin migreringskod som de hade hittat på GitHub, och sedan anpassat. Det gav oss ett försprång med att bygga det vi behövde. Till slut hade vi över 8000 rader kod som migrerade flera miljoner dataposter på under två timmar.

Denna data fyllde mellan 20 och 30 viktiga tabeller, såsom bibposter, exemplar, låntagare, avgifter, auktoriteter, prenumerationer, periodika, i Kohas databas. Den enda data som var riktigt svår att hantera var prenumerationer.

Tekniska utmaningar

För alla, utom för auktoriteter och periodika, extraherades all data från Oracle databasen som hörde till Virtua med hjälp av SQL-frågor inbäddade i dbd-kod. De två undantagen extraherades till marcfiler genom att använda Virtuas exportfunktion. De delar av koden som vi hade från början var skrivna för en gammal version av perl och behövde flera uppdateringar. Koden var inte optimerad för stora mängder data så även små mängder data tog flera minuter och stora dataset tog tiotals timmar. Vi insåg tidigt att migrera data var en uppgift som vi skulle behöva upprepa flera gånger, för att kunna mappa datan från Virtuas proprietära system till Kohas öppna. Vi arbetade således mycket för att optimera processen. Existerande kod korrigerades och optimerades, och vi kunde ofta köra parallella importer för att snabba upp migreringen. Servrar med flera kärnor användes också för att reducera tidsåtgången. Vi kunde arbeta med intermediära tabeller samt filer så vi inte behövde göra totalmigreringar så ofta förrän den slutliga migreringen.

Migrering av prenumerationer

Den stora utmaningen var att migrera periodikan. Vi kunde bara exportera marcfiler från Virtua och Koha behöver tre tabeller populerade med innehållet från 853. Det krävdes enorm parsing av detta fält som visade sig vara väldigt tidskrävande att skriva beroende på stora mängder unika taggar och andra datasvårigheter. Det blev ohållbart så de enda variationer av 853 som togs med var de som användes av flera prenumerationer. De mönster som alltså bara användes av en eller ett fåtal tidskrifter togs inte med.

Testning

Testning var en svag länk, och få bibliotek hade möjlighet att kontrollera resultaten av migreringen för att försäkra att datan var korrekt och konsekvent. Testningen fick i praktiken göras genom att använda Koha med nya datan och kontrollera att Koha uppförde sig som förväntat. Det utfördes dock ett fåtal, systematiska tester för att kontrollera datan hamnat på rätt ställe.

Resultat

Trots svårigheterna att testa behövde väldigt lite data korrigeras när problem upptäcktes efter vi gått i produktion. Prenumerationerna behöver dock fortfarande (tre månader efter lansering) extra support. Koden, i perl5, som användes för migreringen finns tillgänglig för fri användning.

Servrar

Vår Koha-installation ligger i en klustrad, distribuerad och virtuell miljö. Skalbarhet och hög tillgänglighet har varit viktigt för oss. Kohas webbapplikation (frontend) körs på två lastbalanserade webbservrar för redundans, och en tredje server som hanterar cronjob och ej webbaserade funktioner som NCIP. Databasen är en trenavs galera mariadb som också är lastbalanserad för redundans. Sökningarna hanteras av ett fem-nav (två gateway/router, tre workers) elastic search

cluster. Alla dessa körs på virtuella maskiner som ligger på en femnavs kvm kluster[2]. Det är konfigurerat att tillåta alla virtuella maskiner att köra på alla fysiska värdar samt att stödja live migrering. Lagringen av kvm-värdarna tillhandahålls av en replica 2 arbiter gluster fs, för hög tillgänglighet.

Utveckling

Vi behövde utveckla en del funktionalitet innan vi kunde lansera Koha som vårt ILS. Vi hade en kort lista över funktioner som måste finnas innan vi tog systemet i drift. Andra saker kunde vänta till senare.

Beställningar

Det första vi saknade i Koha, var en beställningsfunktion. Eftersom vi är mycket beroende av en välfungerande sådan funktionalitet, var det något av det första vi tog itu med. Vi har olika typer av material, med olika användningsområden som ska kunna hanteras av samma system. Det är allt från efterfrågad kurslitteratur till tidskrifter till gamla böcker som inte får lämna huset. Vidare har vi både öppna hyllor och slutna magasin, och det finns 26 olika bibliotek i vår organisation. Vårt gamla system var väldigt moget och utvecklat, och kunde tillgodose de flesta av våra krav på funktionalitet, så det behövde vi kunna ersätta. Ganska snart vi fick till en funktion som vi kunde visa upp. Det inkom synpunkter på våra reviewer, och när vi i januari 2018 gjorde en användbarhetstest på Koha så långt som vi hade kommit då, förstod vi att vi behövde göra ganska stora förändringar för att få en lättbegriplig beställningsfunktion. Efter diverse justeringar fick vi till vad vi tyckte var ett bättre flöde än det vi hade i tidigare system, även om det kvarstår några förbättringar som vi inte hann få på plats före lansering.

Låntagarregistrering

Något vi upplevt som tidskrävande och lite krångligt i vårt gamla system var låntagarregistrering. För ett par år sedan när vi började använda campuskortet som lånekort, började vi importera data från campuskortets databas. Vid systembytet såg vi ett bra tillfälle att vidareutveckla den delen. Vi har två grupper av användare; dels studenter och anställda vid LU, dels tredje man. Vi beslöt att hantera dessa olika och att fokusera på studenterna som är den stora gruppen användare. Vi ville undvika att ha ett formulär som ska fyllas i, och vi kunde inte, på grund av nya dataskyddslagen, GDPR, hämta uppgifter om alla studenter i förberedande syfte. Vi ville också använda oss av den centrala inloggningen för att studenterna skulle ha så få inlogg som möjligt. Dessa förutsättningar gav oss ideer till en lösning. Alla nyintagna studenter får ett centralt id med tillhörande lösenord av LU. Detta används till diverse tjänster. Genom att länka Kohas inloggning till universitets shibboleth-inloggning kunde vi den vägen hämta deras data. Om de inte redan har ett konto i bibliotekssystemet när de loggar in, skapas ett konto åt dem i inloggningsögonblicket. Allt som sedan behöver göras för att de ska kunna låna böcker, är att tilldela dem rätt låntagarkategori och rätt hemmabibliotek i disken. Med denna lösning fick vi minimal arbetsinsats för både studenter och personalen i disken. Vi matchar varje morgon alla studenter mot det centrala registret för att få med förändringar, till exempel om någon byter kortnummer eller personnummer (vilket sker dagligen). Det finns också en

[2] Kvm=kernel based virtual machine

funktion för att hämta den informationen direkt, genom att trycka på en knapp i låntagarposten, om en student precis har uppdaterat sitt LU-konto, och har ett ärende på biblioteket omedelbart efter.

Externa användare, som inte har någon koppling till LU, får registreras manuellt. Deras id skannas, och matchas mot folkbokföringen (Navet). Lånekort och e-postadress får läggas till genom skanning och inskrivning, så de kräver lite mer hantering. De är dock färre till antalet, och är inte mer tidskrävande att hantera än på det gamla sättet, som dessutom innebar mer arbete för användaren själv.

Fakturering

Det saknades också en funktion för fakturering av förlorade exemplar i Koha. Vi fakturerar låntagare som inte återlämnar lånade böcker i tid. Det är ett flertal låntagare som är aktuella för faktura varje vecka, och vi behöver ha bra systemstöd för att kunna hantera det. En lista med låntagare som har så pass försenade böcker att de kvalificerar för faktura, skickas varje vecka till respektive bibliotek. I samband med att detta jobb körs, hämtar vi låntagaradress från folkbokföringen, dels för att vi inte sparar adresser i vår databas, men också för att få den mest aktuella adressen. Adressen skickas med övriga uppgifter för att biblioteken ska kunna förbereda fakturor som skickas per post. I de fall det handlar om utländska studenter, hämtar vi adressen från Ladok. För fjärrlånebibliotek sparar vi adresser i vår databas.

Externa system

Vårt bibliotekssystem behöver data från andra system, och det skickar data till andra system. Libris är förstås ett av de systemen. Samma vecka som vi stängde vårt lokala system, stängde Libris av Voyager, och påbörjade skiftet till Libris XL. Vi beslöt att sitta i båten och ta ett system i taget, och vi började med det lokala. Det visade sig att det vi tagit fram tidigare under våren inte gick att använda rakt av, utan en del justeringar behövde göras. Cirka en månad efter lansering av Koha började vi med de dagliga batchimporterna från Libris.

Vi har ett egenutvecklat fjärrlånesystem, Basill, och detta behövdes kopplas ihop med Koha. De pratar ncip med varandra och det var relativt okomplicerat att få det på plats. Bokomaterna pratar SIP och de har varit lite bökigare. De fungerar att använda med Koha, men det har varit lite inkörningsproblem, bland annat har vi behövt öka time out-tiden, vi har haft datumformatfel och för att kunna använda automaterna vettigt i vår miljö, krävdes lite extra konfiguration gällande utlån av titel med kö (vilket i vissa fall tillåts, i andra inte).

För att underlätta registrering av låntagare ville vi hämta data från LU-centrala system. Låntagarposter synkas med data från LUs system, där vi bland annat hämtar e-postadress, campuskortnummer och PIN-kod för att kunna nyttja campuskortet som lånekort. Om det skulle behövas kan vi utöver de dagliga matchningarna hämta information genom ett knappklick, exempelvis om en student har ett nytt campuskort och behöver ha sin låntagarpost uppdaterad direkt. Vi lagrar inga adresser i vår databas, utan hämtar vid behov, en gång i veckan då vi tar ut information om försenade böcker som är dags att fakturera, från Skattemyndighetens tjänst Navet.

Studenters låntagarposter skapas, som nämnts tidigare, med hjälp av protokollet Shibboleth som hämtar aktuell data från våra centrala system, när studenten autentiserar sig via Shibboleth (som används inom LU).

Vid lansering hade vi inte kopplingen till vårt discoverysystem, Ebsco, på plats. Vi räknar med att få det på plats under hösten. Vi ska ha dagliga överföringar av nya, förändrade och borttagna poster. Vi ska också visa tillgänglighetsinformation direkt i Ebsco.

Vi har en egen webbsida där vi kan administrera beställningar, fakturor och annat. Den pratar med bibliotekssystemet, och den fungerar nu med Koha istället för med Virtua.

Testning

En bit in i projektet insåg vi att vi inte skulle hinna med att administrera de tester vi hade för avsikt att göra med hjälp av våra kollegor i nätverket. Vi lyckades knyta en person till projektet som hade som ansvar att driva testerna samt att informera om nyheter och annat. Det var väldigt lyckat att få en bibliotekarie (dessutom med ux-perspektiv) som kunde fokusera på tester, och som inte heller tillhörde den innersta kretsen och därför kunde ställa de bra frågorna innan testarna började sitt arbete. Hon hade helt enkelt lite distans och det har varit bra att ha sådana personer involverade i projektet. Vi efterlyste frivilliga testare i nätverket och fick många intresserade. Dessa delades in efter intresseområde, fördelat på cirkulation, katalog, opac, fjärrlån och periodika. Vår avsikt med testerna var både att testa av funktionalitet och att få in feedback på den, men även att ge bibliotekarierna en chans att bekanta sig med systemet. Testerna gav bra input, och dessutom fick vi in synpunkter på sådant vi inte frågat efter, vilket är både bra och dåligt. Ett nytt system kan vara överväldigande att hantera, både för användare och administratörer!

Vi försökte ha saker att testa i varje sprint, men det var inte alltid vi lyckades med det. På de publika reviewerna ombads vi ibland att visa olika funktioner eller handgrepp, och det blev också ett tillfälle att fånga upp synpunkter och konstigheter.

Testarna gavs konton i Redmine, vårt projektverktyg. Med dessa konton kunde de logga in och registrera ärenden i form av projektrapporter. Det var väldigt bra att få rapporterna direkt i verktyget, så att alla i projektgruppen kunde läsa de rapporter de var intresserade av. Utifrån rapporterna gjorde vi user stories till framtida sprintar eller inlägg till vår FAQ, så att vi kunde fånga upp frågor så snart som möjligt för att ha en välfylld frågebank när vi lanserade.

UX

Utöver testerna med personal, hade vi även en användbarhetstest med studenter. Vi testade vår egenutvecklade beställningsrutin. Testpersonerna fick några uppgifter att utföra. Skärmen spelades in och visades upp tillsammans med ljud i ett annat rum där några ur teamet kunde se var eventuella

problem uppstod. Testerna visade att vi behövde förändra en hel del i visningen av exemplar, och även lite i hur själva beställningen gick till. Tyvärr hann vi inte med fler tester innan vi gick live. Några månader efter lansering gjorde vi dock ett liknande test, och hittade andra saker vi behöver jobba vidare på. Användbarhetstester är något vi kommer arbeta med regelbundet framöver.

Utbildning

Tidpunkten för att starta med utbildning, var inte självklar. Vi arbetade med utveckling av Koha ända in i det sista, och mycket av gränssnittsförändringarna hade vi sparat till sist. Skulle vi börja tidigt med utbildning för att ge oss alla gott om tid till förkovring, och då visa upp ett system som skulle förändra sig på flera sätt innan lansering, eller skulle vi vänta längre för att kunna ha utbildning på ett system som skulle se likadant ut i produktion? Vi satte igång med utbildningen i mitten av april, och hade flera tillfällen under april och maj. Utbildningen fokuserade i första hand på de handgrepp som används i disken. För att utbildningen inte skulle påverkas av utveckling och förändringar som gjordes på våra test- och utvecklingsservrar, gjorde vi en speciell klon för utbildning. Vid utbildningstillfällena kom också synpunkter och frågor, och vi antecknade för att lägga in önskemål på framtida utveckling eller funktionalitet, eller i förekommande fall publicera en fråga i vår FAQ.

FAQ

Den undergrupp som arbetade med utbildning skrev också en FAQ. Vi hade inte tid att skriva några manualer utan la tiden på att utveckla en FAQ istället. Vi använde frågor som kom in i våra olika kanaler såsom utbildning, reviewer och tester. Vi frågade också efter Göteborgs UBs vanligaste frågor i samband med lansering. De bytte från samma system till Koha några månader innan vi, och var snälla att dela med sig av de frågor de fick i början av tiden med Koha. FAQerna har varit till nytta för våra bibliotekarier, kanske framför allt de som bemannar vår virtuella disk, till vilken en mängd frågor inkommit sedan vi drog igång Koha. Utöver den interna FAQen, har vi även gjort en publik FAQ, som finns både i en svensk och en engelsk variant.

Lansering

Den 7 juni stängde vi vårt gamla system. Vi hade redan den 24 maj stängt av möjligheten att beställa våra böcker, för att minimera risken för att en bok stod för avhämtning under perioden vi var utan system. Efter avstängningen påbörjades arbetet med skiftet. Vi gjorde först en klon av vår databas som den såg ut när vi hade stängt ner systemet. Klonen tillgängliggjorde vi senare via vår gamla testmiljö, så den kunde användas av bibliotekarierna för referens, till exempel för att kontrollera låntagare och annan information som inte finns via Libris. Sedan började migreringsarbetet. Vi migrerade till produktionsmiljön, varefter vi verifierade och startade indexeringen. Därefter kopierades datan till vår test- respektive utvecklingsmiljö. Eftersom Libris valde precis samma dagar att skifta till Libris XL, var de dagliga Librisimporterna inte klara för laddning i samband med lansering, utan Librisposterna fick vi vänta med. Vi hade förberett så att det skulle vara klart att köra

igång, men det dröjde till slutet av juli innan vi hade importen på plats. Det berodde både på kvaliteten på Libris-posterna, men också på finjusteringar av importen på vår sida. När den mest intensiva delen av migrering och konfigurering var klar, hade vi dagliga avstämningsmöten i projektgruppen, där vi gick igenom utestående buggar och fel. Mötena avsåg att avgöra om systemet var redo för lansering, eller om vi behövde åtgärda något först. Vår Koha-installation var inte perfekt vid tidpunkten, men helt klart funktionsdugligt. Det testades före lansering av en grupp testare, och vi upptäckte en del mindre misstag. En del kunde vi åtgärda, andra fick vi leva med. Några misstag upptäcktes inte förrän i produktion. Exempel på dessa var att vi hade kastat om två kolumner i våra beställningar, vilket innebar att viss information gick förlorad. Vi kunde korrigera det hjälpligt genom att göra om exemplarbeställningar till bibliografiska beställningar. Några 100 låntagare fick fel lånekort migrerat (på grund av att de var felaktigt makulerade i gamla systemet). Det gick dock bra att korrigera. Det var också en del bekymmer med utskrifter orsakade av popup-rutor, webbläsare, och att vi av misstag skrivit ut låntagarid istället för lånekortsnummer på reservationslapparna. Blandade fel med andra ord. Den 19 juni öppnade vi systemet för våra användare.

Vi hade förberett vår informationstjänst med ökade öppettider, och försökt ha många svar beredda. Vi utgick från att de frågor vi fått under projekttiden skulle komma, liksom en del av de frågor våra kollegor i Göteborg hade fått. Det inkom flera frågor i början av driftsfasen, avtog lite under semestertiden, och ökade igen när terminsstart närmade sig.

Driftsfas

Den 19 juni gick vi live med LUBcat som vi kallar vårt bibliotekssystem. Vi ville göra bytet på sommaren, och så tidigt som möjligt för att ha gott om tid att fixa fel och buggar innan terminsstarten skulle dra igång. Vi skulle också få tid till personalen att känna på systemet under den minst stressiga tiden på året, och dessutom ge utrymme för projektgruppen att hålla lite semester. De dagliga avstämningsmötena, utöver de dagliga scrummötena, fortsatte efter lansering. Vi samlade fel och buggar i en lista i vårt projektverktyg, och några ur projektgruppen bedömde allvarlighetsgraden på dessa och la in det antingen i aktuell sprint eller i en framtida dito.

I driftsfasen arbetar 7 personer (IT-avdelningen) med Koha. Av dessa är det en person som i nuläget arbetar heltid med Koha. Övriga fördelar sin tid även på andra projekt och system, samt på allt från ordinär IT-support till serverunderhåll.

Vi gick live med en Koha-version från januari 2018, dvs 17.11-1. Den innehåller cirka 600 tillägg, så vi kan lugnt säga att vi divergerat en del från community-versionen. Användandet av Elastic search har krävt en hel del anpassningar, förutom det har vi beställningsfunktionen som har krävt sitt. Vi har gjort en del översättningar, varav somliga har rapporterats till den svenska allmänna översättningen. Så det är allt från stort till smått som vi lagt till för att få det ILS vi önskade. Vi kommer fortsätta med egna tillägg, och hoppas kunna få in en del av våra tillägg i community-koden. Vi har ännu inte bestämt hur vi ska arbeta vidare med uppdateringar från communityn, när vi väl vill uppdatera har vi många tillägg att hantera. Givetvis vill vi ha rättningar och ny funktionalitet men det får vägas mot

merarbetet att lägga på våra anpassningar på en ny Koha-version. Framför allt är vi intresserade av utveckling för att stödja Elastic search, då det kunde eliminera en del av våra egna anpassningar.

Det som har varit svårast, både för personal och projektgruppen, har varit periodikahanteringen. Virtua var väldigt utvecklat på den fronten, sedan många år. Det var någorlunda buggigt, men fungerade väl för våra behov. Det hade också en väldigt avancerad marcstruktur som antagligen inget system kan matcha. Det innebar utmaningar vid migreringen, och har även inneburit förändrade rutiner eftersom all data inte kunde migreras på ett vettigt sätt. Vi la ändå väldigt mycket tid på att få med så mycket vi kunde till de bästa ställena vi kunde komma på. En del fick vi hoppa över, till exempel hade vi onödigt många mönster sparade i gamla systemet, och de mönster som bara använts en eller ett par gånger tog vi inte med oss. Utöver delvis nya rutiner för registrering av nya häften, är visningen sämre i Koha än i Virtua. Det får vi leva med, och lära oss på nytt. Fördelen med Koha mot Virtua är ändå enkelheten; ibland kan hög flexibilitet innebära onödigt krångel, både för användaren och administratören.

I övrigt upplever vi att personalen har varit snabba på att lära sig det nya systemet. Det kommer fortfarande in många frågor, men de har ändrat karaktär, och liknar allt mer de frågor som kom in på det gamla systemet. Det återstår förstås en hel del att göra, innan vi har ett system vi kan känna oss nöjda med. Det finns några övergångsproblem som kommer fasas ut när alla transaktioner är gjorda i det nya systemet. Som exempel kan nämnas att vi i det tidigare systemet använde långlån (öppen lånetid som förkortades vid kö). Dessa kunde inte översättas exakt i Koha, utan uppför sig något annorlunda än andra, kortare lån.

Vi uppdaterar vår produktionsmiljö varannan vecka. Det arbete som görs i en sprint görs på en utvecklingsserver, och läggs när sprinten är slut på vår testserver. Där har vi två veckor på oss att testa, och om allt fungerar tillfyllest, kommer det in i produktion efter dessa två veckor. Och så börjar det om igen. Är något av allvarligare karaktär, gör vi en hotfix och lägger på den direkt. Det har än så länge bara använts i några fall.

Genom att använda Redmine som projektverktyg, kombinerat med ett visst mått av disciplin har vi lyckats att få ganska bra dokumentation av projektet och av systemet. I Redmine finns bland annat en wikidel, som vi använt till att dokumentera allt från förarbetet, till projektbeslut, instruktioner och how-tos, anteckningar från reviewer och retrospective.

LUBcat driftas enligt förvaltningsmodellen pm3. Pm3 används av flertalet system inom LU.

Vidareutveckling

Under sommarperioden har vi mestadels ägnat oss åt att hantera de uppgifter som vi fick prioritera ner inför lanseringen. Vi hann inte med allt vi hade i vår backlog, och mot slutet av projektperioden

fick vi prioritera hårt. Men efter semestrarna kan vi nu börja tänka på nya saker. Projektgruppen kommer att minska i mantimmar, men vi har fortfarande många uppgifter att utföra. Vi önskar att få till en bättre betalningslösning av förseningsavgifter, och det står högt på önskelistan. Vi hade till vårt gamla system en desideratalista, och den valde vi bort att ha på plats vid lansering. Den ska tillbaka. Sedan finns det mindre förbättringsförslag som inkommit under sommaren, och själva har vi idéer kvar som vi hoppas kunna förverkliga. Men först ska vi se till att det nödvändiga fungerar som det ska, så alla kan sköta sina arbetsuppgifter på ett någorlunda tillfredsställande sätt.

Vi bytte från ett server/klient-system till ett webbaserat system, och upplevde att det nya systemet var väldigt långsamt. Vi har jobbat med att snabba upp det på olika sätt, bland annat har vi lagt på webbservern Starman för att få upp hastigheten. Det återstår dock en del saker vi kan finjustera för att förbättra snabbheten.

Vi har fått tillfälle att utvärdera vår databaslösning, efter ett antal kraschar av olika grad under hösten. I skrivande stund planerar vi att överge galera-lösningen för en master/slave-variant.

Erfarenheter

Det föll sig så att LIBRIS bestämde sig för att byta till Libris XL samma vecka som vi gjorde skiftet mellan våra system. Det gjorde att det tog lite längre tid innan vi fick igång importerna, eftersom vi behövde koncentrera oss på att få igång vårt nya ILS först. Det var dock inget att göra åt, vårt skifte var inplanerat sedan länge med allt vad det innebär av överenskommelser med externa personer och kommunikation till användare. Att vi valde juni som övergångstid visade sig vara ett klokt val; det gav oss ganska lång tid efter bytet då det var mindre aktivitet i systemet, då vi både kunde lära oss systemet under en period som var mindre hektisk än under terminstid. Vi kunde också rätta buggar och andra fel när pressen var lite lägre än normalt. Dessutom kunde nästan alla i gruppen hålla semester nästan som vanligt under sommaren, vilket vi alla behövde.

Speciellt för katalogisatörerna blev det besvärligt, att förstå om ett fel härrörde ur Koha eller LIBRIS XL, eller från importen av data. Och att lära sig två nya system samtidigt är inte att föredra.

Att lära sig ett nytt system tar tid, även för systembibliotekarier och systemutvecklare. Det ska inte underskattas. Även om det kan vara enkelt att installera ett program och fylla en databas med innehåll, så räcker inte det hela vägen. Konfigurationsmöjligheterna är stora, vikten av att få rätt data i rätt format på rätt plats är avgörande. Och ett nytt system beter sig nödvändigtvis inte på samma vis som andra system. Kanske kommer folk in med förutfattade meningar om hur saker och ting ska fungera, allt från utvecklare och systembibliotekarier till bibliotekarier i disken.

En sak vi insåg, tydligt, när vi närmade oss lansering, och testandet av Koha var inne i en intensiv fas, var att Koha inte är designat för större bibliotek. Vi har låntagare med väldigt många lån, och innan vi hade vidtagit åtgärder, gick det inte att visa lånen för en låntagare som hade fler än 500 lån för att

systemet timeade ut. Vi behövde skala bort en del kontroller Koha gjorde i samband med visningen av lån, samt att låta bli att visa undertitel, och sedan gick det att få fram lånen i en dylik post.

Att ha en hård deadline, och att jobba tätt och intensivt i en grupp under så pass lång tid, kan också innebära sina utmaningar. Det har varit god stämning i gruppen mestadels, men periodvis har det varit stor stress och vi har inte alltid varit överens om hur saker ska göras. Meningsutbyten har inte saknats, det har varit stressigt och svårt, men det har aldrig varit tråkigt.

När det gäller metoden, scrum, lärde vi oss väldigt snart, i teorin, att nyckeln till framgång var att skriva korta och tydliga user stories och taskar (arbetsbeskrivningar). Det gick snabbt att inse, däremot har det varit svårt att omsätta i praktiken. För att en user story ska bli gjord, ska den utgöras av flera små uppgifter, som är enkla att förstå. Det som kan tyckas självklart när uppgiften skrivs, kan vara ett mysterium ett par veckor senare, när någon annan än författaren av uppgiften ska utföra den. Målet är att vem som helst i gruppen ska kunna ta uppgiften och förstå innebörden. Eftersom vår projektgrupp har bestått av personal med ganska olika kompetens, har dock flera av uppgifterna varit riktade till en specifik person. Somliga har också behövt skriva sina egna uppgifter, eftersom ingen annan i teamet har haft tillräcklig kunskap för att ens förstå att en uppgift behövs göras (till exempel felhantering av databaskluster). Det är alltså viktigt att alla känner att de kan skriva user stories och att de behövs för att föra projektet framåt. En annan sak som vi tidigt ifrågasatte i vår scrummetod, var att logga tiden. Det upplevdes som svårt, och vi övergick till att räkna i story points istället. Story points tilldelas uppgifterna, efter Fibonacciskalan[3], och det har gett oss ett verktyg att bedöma uppgifternas svårighet och uppskattad tidsåtgång. Många är gångerna då vi felbedömt, men överlag har det varit ett värdefullt hjälpmedel för oss att räkna ut hur många uppgifter vi ska ha med i en sprint. Antalet uppgifter i en sprint beror då på hur många story points de sammanlagt har. Vi kunde också göra en ungefärlig uppskattning över tiden vi hade kvar fram till lansering, genom att räkna story points och se hur många vi statistiskt skulle klara av att göra varje sprint. Det blev likt ett verktyg som hjälpte oss att inse att vi behövde prioritera ner somliga uppgifter till efter lansering.

Enligt en del metodböcker om scrum ska en uppgift också innehålla en klardefinition. Det vill säga att det ska vara tydligt formulerat i uppgiften vad som krävs för att räkna en uppgift som klar. Vi använde oss inte av det, även om vi ofta pratade om att det nog hade varit bra. Speciellt under perioden före lansering hade det varit hjälpfullt för oss att ha det klart formulerat. Vi hade exempelvis flera olika servrar och det var inte alltid tydligt var något skulle göras. I och med att vi kom in i produktionsfas, har vi en tydlig release-cykel och det är mer självklart var något ska göras. Det finns dock andra omständigheter som kan göra en uppgift tydlig, vi har alla olika bakgrund, och det som är glasklart för en är en gåta för en annan.

Vi pratade under hela projekttiden om att vi önskade använda oss av automatiska tester. Vi undersökte ett par olika möjligheter att få igång det, men på grund av tidsbrist blev det nedprioriterat. Det är dock något vi så här i efterhand önskar att vi fått tid till, och vi kommer försöka få det på plats framöver.

[3] https://sv.wikipedia.org/wiki/Fibonaccital

Sammanfattning

När vi från början fick uppdraget att byta biblioteksystem, med beskedet att det fick ta ett år, kände flera av oss stor tveksamhet. Inte kan vi väl bli klara med detta gigantiska projekt på bara ett år? Vi gjorde ett ärligt försök, och hade målbilden tydlig för oss. Vi skulle leverera ett nytt system sommaren 2018. Det var aldrig självklart att vi skulle hinna, men det gjorde vi. Hade vi haft mer tid på oss, hade vi kunnat leverera ett mer färdigt system, men det vi levererade var helt klart funktionsdugligt. Det är inte farligt att sänka sin ambitionsnivå, så länge den inte hamnar alldeles för långt ner. Självklart ska systemet kunna användas, och någorlunda motsvara det tidigare systemet. Eftersom vi arbetat med agil utveckling har vi kunnat justera saker efterhand, och om vår publik på reviewerna har påpekat saker som vi kunnat åtgärda, har vi gjort det. Nu efter lansering kan vi åtgärda det som kvarstår, om det fortfarande är aktuellt. Ibland behöver man bara anpassa sina rutiner något, och det nya sättet att arbeta på kanske inte alls är så mycket sämre än det gamla. Förstås kommer det in nya frågor kontinuerligt, och vi prioriterar, om inte dagligen, så ändå regelbundet, bland önskemålen och förbättringsförslagen. Det gäller att hålla huvudet kallt, och se vad som gynnar hela vår komplexa biblioteksorganisation, så ärendena betas av i rätt ordning. Vi räknar med att ha mycket att göra åtminstone ett halvår framöver, sedan får tiden och resurserna avgöra vilken nivå vi hamnar på gällande utveckling. Men en sak är säker: ett biblioteksystem blir aldrig färdigt.

Bilaga 1

Scrum team: Björn Nylén, Dave Sherohman, Frank Hansen, Jan Gustafsson, Kevin Carnes, Martin Persson, Snorri Briem

Scrum masters: David Holoshka, Maria Hedberg

Product backlog owner: Stina Hallin

Testansvarig: Ingela Wahlgren (till mars), Åsa Forsberg (från mars)

Utbildningsansvariga: Martin Persson, Maria Ohlsson, Therese Ericsson